BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
12 April 2001 (12.04.2001)

PCT

(10) International Publication Number
WO 01/25570 A1

(51) International Patent Classification[7]: E05B 49/00

(21) International Application Number: PCT/US00/21989

(22) International Filing Date:
29 September 2000 (29.09.2000)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/157,152 30 September 1999 (30.09.1999) US

(71) Applicant: MAS-HAMILTON GROUP, INC. [US/US]; 805-D Newtown Circle, Lexington, KY 40511 (US).

(72) Inventors: DAWSON, Gerald; 4942 Pelican Drive, New Port Richey, FL 34652 (US). EDWARDS, Robert, L.; 1930 Natchez Trail, Lexington, KY 40504 (US). REED, Tim; 325 Chamberlain Drive, Lexington, KY 40517 (US). SCHAEFER, Adrian; 10910 Layton Street, Largo, MD 20774 (US). WOLTERMAN, David; 2295 Harrods Pointe Trace, Lexington, KY 40514 (US).

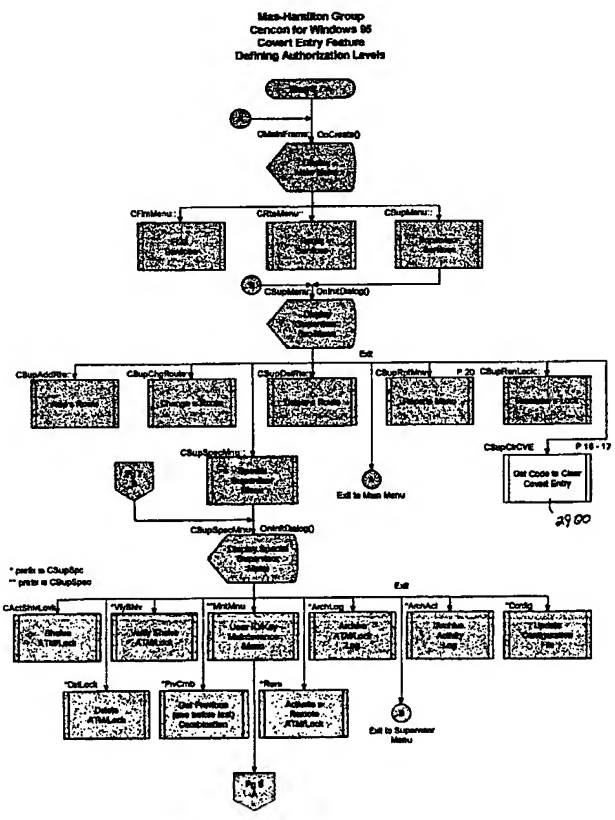
(74) Agents: HILL, Rustan, J. et al.; Arent Fox Kintner Plotkin & Kahn, PLLC, 1050 Connecticut Avenue, NW, Washington, DC 20036-5339 (US).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian

[Continued on next page]

(54) Title: METHOD AND APPARATUS FOR USING A ONE-TIME COMBINATION OR CODE AS A RESET COMBINATION FOR A COVERT ENTRY FEATURE ON A LOCK



(57) Abstract: The present invention provides a method for resetting the covert entry lockout feature of an electronic lock with a one-time reset combination. The one-time reset combination may also permit opening the lock. There is a reset combination. This combination cannot be utilized to reset the covert entry feature of a particular lock two times in a row. The lock operator enters the combination into the lock. Thereafter, the locks microprocessor or electronic control system verifies that the entered combination is authorized to clear the covert entry by comparing the autorized and entered reset combinations. When the entered reset combination compares equal to the authorized clear/reset combination, then the covert entry feature would be reset.

WO 01/25570 A1

patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:

— *With international search report.*

— *Before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments.*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

METHOD AND APPARATUS FOR USING A ONE-TIME COMBINATION OR CODE AS A RESET COMBINATION FOR A COVERT ENTRY FEATURE ON A LOCK

BACKGROUND

Field of the Invention

This invention relates to the field of devices that detect covert opening or attemped covert opening of electronic locks, and more particularly, to the use of a one-time combination or code that is utilized as a reset combination or reset and opening combination to reset an electronic lock that is disabled after the electronic lock detects the covert opening and/or an attempted covert opening.

Description of the Related Art

Some electronic locks, including the CEX-07 manufactured by the Mas-Hamilton Group, are capable of detecting either the covert opening or an attempted covert opening of the lock. Typically, these locks prevent an individual with an authorized combination or access code from opening of the electronic lock once a covert opening or an attempted covert opening has taken place. The holders of an authorized combination are in effect "locked out" until the covert entry feature is reset or bypassed.

To reset a covert entry electronic lock; a covert entry reset combination is employed. This reset combination, however, does not usually change

unless the individual with the reset combination manually changes the reset combination. Typically, this reset combination is changed in the same manner that the regular combination has changed. Therefore, to maintain proper security over the reset combination only one individual or a small group of individuals, typically supervisory personnel, know the reset combination. It is practical to use a single individual or a small group with the reset combination, particularly when there are a limited number of locks and/or those locks are located in the same general area.

It is not cost-effective or practical, however, to limit the reset combination to an individual or a small group with the employment of one-time combination locks serviced over a wide geographical area. The disclosure of the reset combination to any individual who required the reset combination to open a lock that had experienced a covert entry or attempted covert entry, however, would compromise the security of all locks utilizing that particular reset combination, until the reset combination on all locks utilizing that reset combination could be changed.

Therefore, a one-time reset combination is needed that will permit the lock to be reset or reset and opened by an authorized individual and which will also notify supervisory personnel that a particular container's lock may have been covertly opened. Additionally, there is also a need for a lock to be able to indicate when the lock subjected to tampering while open.

SUMMARY OF THE INVENTION

The present invention solves the problems discussed above and provides a method for resetting the covert entry lockout feature of an electronic lock with a one-time reset combination. Preferably, the one-time reset combination will also permit opening the lock. There is a reset combination. This combination cannot be utilized to reset the covert entry feature of a particular lock two times in a row. The lock operator enters the combination into the lock. Thereafter, the locks microprocessor or electronic control system verifies that the entered combination is authorized to clear the covert entry by comparing the authorized and entered reset combinations. When the entered reset combination compares equal to the authorized clear/reset combination, then the covert entry feature would be reset.

In the preferred embodiment, the reset combination would also be checked to see if that combination was also authorized to open the lock. When the entered combination compares equal to the opening combination, then the lock would also be enabled for opening. Alternatively, a second combination could be entered to open the lock.

DESCRIPTION OF THE DRAWINGS

The accompanying drawings incorporated in and forming part of the specification illustrate several aspects of the present invention, and together with the description serves to explain the principles of the invention. In the drawings:

Figure 1 illustrates the main menu that may be displayed after the lock management software is started.

Figure 2 illustrates the supervisor services menu that may be called from the main menu shown in Figure 1.

Figure 3A illustrates the covert entry clear code menu accessed from the supervisor services menu shown in Figure 2.

Figure 3B illustrates a second example of the Get Code to Clear Covert Entry dialog box shown in Figure 3A.

Figure 3C illustrates the Dispatch a FLM Services Call dialog box.

Figure 4A illustrates an exemplary window that displays the covert entry reset code.

Figure 4B illustrates a second example of the clear code message box shown in Figure 4A.

Figure 5 illustrates the FLM services menu that is accessed from the main menu shown in Figure 1.

Figure 6 is the route services menu that is accessed from the main menu shown in Figure 1.

Figure 7 is an exemplary window utilized to close a service call that is accessed from either the FLM or route services menus of Figures 5 or 6.

Figure 8 illustrates an exemplary message window displaying a covert entry occurred when closing a service call utilizing the Closing a Service Call window shown in Figure 7.

Figure 9 illustrates an exemplary Close a Route Using Menu window that is accessed from the route services menu shown in Figure 6.

Figure 10 illustrates an exemplary window that may be utilized to Enter Close Codes for a Route that is accessed using the close a route using a menu of Figure 9.

Figure 11 illustrates an exemplary Close a Route Using Keys window that may be accessed from the routes services menu of Figure 6.

Figure 12 illustrates an exemplary message window that may be displayed when a cover covert entry was reported when the route was closed using the window shown in either of Figures 10 or 11.

Figure 13 illustrates the Supervisor Reports Menu that is accessed from the Supervisor Services menu shown in Figure 2.

Figure 14 illustrates the Activity Log that may be accessed from the Supervisor Reports Menu shown in Figure 13 .

Figure 15 illustrates the List SA Key function that is accessed from the Supervisor Reports Menu shown in Figure 13.

Figure 16 illustrates the flow chart index for the portion of software that is utilized to obtain the covert entry clear code.

Figure 17 illustrates the flow chart showing the program logic used to initialize the software program.

Figures 18 - 24, 25A, 25B, 25C, and 25C-1 illustrate software routines that are utilized in the authorization functions of the software program.

Figures 26 A and B, 27 A and B, and 28 A -D illustrate the program logic utilized to define the authorization levels of a particular program function, action, or request.

Figures 29 and 30 illustrate the program logic employed to call the subroutine employed to create a covert entry clear code.

Figures 31A and B illustrate the program logic employed to obtain a covert entry reset combination.

Figures 31A-1 and 31B-1 illustrate a second embodiment of the get code to clear covert entry subroutine.

Figures 32A and B illustrate the exemplary program logic used to list an activity log.

Figure 33 illustrates the program logic employed to display the Supervisor Reports Menu shown in Figure 13.

Figures 34A, B, and C, 35, and 36 illustrate exemplary program logic that is utilized to display the activity log shown in Figure 14.

Figure 37 illustrates the flow chart index for the portion of the software that is used to detect a cover covert entry when a service call is closed.

Figure 38 illustrates a functional overview of the program logic utilized to determine if a cover covert entry has occurred.

Figure 39 illustrates the program logic utilized to close a service call.

Figure 40 illustrates the close lock subroutine called from the program logic shown in Figure 39.

Figure 41A, B, and C illustrates the subroutine used to verify the closed code and close a service call called from the program logic shown in Figure 40.

Figures 42A, B, and C illustrate exemplary programming logic used when closing a route when using a menu.

Figure 43A and B illustrate the display message for each lock tab with an error subroutine called from the programming logic used to close a route using a menu shown in Figure 42.

Figures 44A, B, and C illustrate the close route subroutine called from the close a route using menu programming logic shown in Figure 42.

Figures 45A, B, and C illustrate exemplary programming logic utilized to close a route using keys.

Figure 46 illustrates the exemplary subroutine logic utilized to check for a cover covert entry closed code.

Figure 47A and B illustrate the logic changes inserted into the main line program utilized in the firmware/software utilized in the lock microprocessor.

Figures 48A and B illustrate a new subroutine added that checks for a covert entry on the lock.

Figure 49 illustrates changes made to the firmware/software utilized to save data to nonvolatile memory (EEPROM).

Figure 50 illustrates the firmware/software subroutine changes utilized in the procedure that moves data nonvolatile memory (EEPROM) to RAM memory.

Figures 51A and B illustrate in pseudo code inserts in the combination in subroutine that were made to facilitate entry of a covert entry reset combination.

Figures 52A and B illustrate the changes made to the subroutine that checks for open indicators/flags when the lock is closed in order to check for tampering and/or covert entry.

Figure 53 illustrate the programming logic utilized to create a new (cover covert) close seal in the event that a back cover was removed while the lock was opened.

Figure 54 illustrates changes to the general delay subroutine that adds the ability to check for a covert entry while the routine is running.

Figure 55 illustrates changes made to the combination generating subroutine to generate the covert entry clear combination.

Figure 56 illustrates changes made to the save open subroutine to facilitate the covert entry feature.

Figure 57 illustrates the changes made to the wait for touch memory routine that adds a call to check for covert entry.

Figure 58 illustrates the changes to the process change key subroutine to facilitate the covert entry feature as programmed.

Figures 59, 60, 61A, and 61B illustrate the dispatch a service call subroutine called from the get covert entry code subroutine of Figure 31.

Reference will now be made in detail to the present preferred embodiment of the invention, an example, of which is illustrated, in the accompanying drawings.

DESCRIPTION OF PREFERRED EMBODIMENT

Overview

In the preferred embodiment there is a software package that runs on a computer. This computer may either be set up as a stand-alone system or to operate on/over a network. This software package generates a one-time reset combination and may determine from the close seal entered that the lock's back cover was removed while the lock was open. Preferably, the software also generates the one-time open combination so that a combined reset and open combination may be issued. Thus, the one-time reset combination may reset the lock or both reset the lock and open the lock. If the reset combination both resets and opens the lock, then it is preferred that the one-time open combination be modified to indicate to the lock that the combination is also valid to reset the covert entry feature of the lock.

The lock firmware/software is also modified to recognize the covert entry reset combination. Upon entry of a valid covert entry reset combination, the covert entry feature is reset. Thereafter, the lock may be enabled for opening upon entry of an authorized open combination. In the preferred embodiment, the reset combination would also be an opening combination. Therefore, in the preferred embodiment, the lock compares/checks the reset combination to see if they reset combination was a valid opening combination.

Computer Software

Illustrations of an exemplary graphical user interface are provided in Figures 1 through 15. The flow charts for the computer software are illustrated in Figures 16 through 46. These Flow charts are provided in two sections: (1)

clear covert entry and (2) detect back cover covert entry. Only the additions/changes to the software will be discussed in detail.

The covert entry feature shown in the accompanying drawings and figures illustrates the implementation of a covert entry feature in a one-time combination lock produced by Mas-Hamilton Group, Inc. Consequently, these flow charts show how to implement this feature in the software code utilized by Mas-Hamilton Group. A computer programmer of ordinary skill in the art could utilize the flow chart shown to implement this feature in other lock software.

Clearing a Covert Entry

Figure 16 illustrates the flow chart index for the 9 major portions of the software 100. This figure is provided to aid the programmer in understanding the organization of the flow charts provided.

Figure 17 illustrates the program 100 initialization sequence that results in the main menu shown in Figure 1. The changes to the initialization routine are shown in the initialize authorization levels erray subroutine 102. This change in the initialization program shown would not be required if authorization levels were not utilized in the program. Authorization levels limit the use of some functions to different levels of supervisory authority. The use of authorization levels, however, increases the security of the lock control system and is therefore included in the preferred embodiment. The authorization levels is not part of the current invention. Consequently, this feature is disclosed to meet best mode requirements.

Figure 18 illustrates the initialize authorization levels array subroutine102 called from Figure 17. The changes in this subroutine are

shown in the set default authorization subroutine 104, which is shown in greater detail in Figure 25A discussed below. Figures 19, 20, 21, 22, and 23 illustrate subroutines that are utilized to perform the authorization functions but were not changed to implement the covert entry feature and as such are shown for complete disclosure, but do not require detailed explanation.

Figures 24A and B illustrate the subroutine that is utilized to update the authorization levels when they are changed as discussed below. The changes to this subroutine are illustrated in the build activity log record subroutine 106 and the translate authorization function resource ID to description subroutine 110. Figure 24C-1 illustrates the translate authorization function resource ID to description subroutine 110 and shows the addition of decision block 112 and process block 114 used to implement the covert entry feature. The decision block 112 checks for the function ID used to identify the covert entry feature and thereafter if this function ID is found assigns a character string to that function ID for display other use in block 114.

Figures 25A and B illustrate an exemplary program logic utilized to set the default authorization levels and build the authorization levels erray. The process block 108 was added to this subroutine to add the get code to clear covert entry function to the authorization level erray so that this array includes the covert entry feature.

Figures 26A and B illustrate the program overview that allows a user to get from the main menu shown in Figure 1 to either the Get the Code to Clear Covert Entry subroutine 2900 or to the Supervisor Service menu subroutine 122 that allows one to change the authorization level required to access or employ a particular function. After the software 100 has initialized the main

menu 10 shown in Figure 1 may be displayed. Typically, the user selects or clicks or selects access to the supervisor services menu selection button 12. The Supervisor Services Menu 14 may be displayed as shown in Figure 2. This menu may provide access to the Special Supervisor Menu through button 13, access to a Reports Menu button 15 and/or a Get Code to Clear Covert Entry button 16. If the user selects the Special Supervisor button 13 then the user, if authorized, would be able to select from the Special Supervisor Menu the Define Function Authorization Levels and thereby change the authorization level required to access various functions.

Figures 27A and B illustrate the programming logic utilized to display the dialog box used to display and change the authorization levels for a particular function. Blocks 124 through 140 shown in Figure 27B illustrate the additions to the subroutine that displays the dialog box to incorporate the covert entry function. As is readily apparent this is just a duplication portion of the existing programming and altering for the clearing covert entry function.

Figures 28A, B, C, and D illustrate the program logic employed when closing and/or clicking o.k. to exit the dialog box that displays the authorization levels for various functions. The changes required for adding the covert entry feature are shown in blocks 142 through 172. As is readily discernable these are just a modification of the existing programming to add the covert entry feature/function.

Figure 29 illustrates the programming logic activated when the user selects or clicks the Get Code to Clear Covert Entry button 16 shown in Figure 2. This program logic is started after the user activates the Get Code to Clear Covert Entry button 16 in input block 2902. Thereafter, a message is sent to

the main frame message map at block 2904 and the subroutine returns at terminator 2906.

Figure 30 illustrates the main frame message map additions to integrate the covert entry clear code/combination feature. Decision block 3202 checks to see if the Get Clear Code to Clear Covert Entry button was selected. When this button and/or function is selected the program logic calls the get code to clear covert entry subroutine 3100.

Figures 31A and B illustrate the program logic utilized in the get code to clear covert entry subroutine 3100. This subroutine begins at start terminator 3102. Thereafter the dialog box is set up in block 3104 and the route data boxes disabled in block 3106. Thereafter a dialog box is displayed in display block 3108. Two examples of the Get Code to Clear Covert Entry dialog box 18 are illustrated in Figures 3A and 3B. After the Get Code to Clear Covert Entry window 18 is displayed the dispatcher and/or supervisor would enter the ATM/lock name in text box 17 and select the lock mode in box 19.

Upon selecting the lock mode the program logic would continue to decision block 3110 to determine if the route mode was selected. When the route mode is selected the route data boxes on the panel are enabled in block 3112 and the program logic returns to block 3108 to update the display. This provides an opportunity for the dispatcher and/or supervisor to enter route data in boxes 21 and 23. Decision block 3114 checks or tests to see if the FLM mode is selected in lock mode box 19. If neither the FLM or Route mode are selected, then the program logic would display an error in block 3116.

When FLM mode was selected in decision block 3114 the route data boxes would be disable in block 3118 and the program flow would return to

display the panel in block 3108. If the route mode or R mode was selected then the user would enter the appropriate data in the route name text box 21 and in the ATM/lock sequence text box 23. Selecting the process button 20, typically generates the clear covert entry combination. Alternatively the user could selected the exit button to close the dialog box and return to the main line program or the lock description button which looks up the data on the lock from the lock table using the following program logic.

Preferably, the subroutine/function called by selecting/clicking the process button 20 obtains data from the panel in block 3120. Thereafter, the routine checks for a lock name in decision block 3122. If a lock name was not entered, then an error message would be set up in block 3124 and displayed at block 3128. If a lock name had been entered in decision block 3122 then the lock mode would be verified using the verify lock mode subroutine 3190 discussed below. Thereafter the program would verify that the lock mode was o.k. in block 3126. Any errors may be displayed at block 3128. When the lock mode was o.k., then the program logic may permit the changing of the lock description in block 3130. Thereafter the lock description would be displayed by using the display lock subroutine 3122. The program flow may then return to display the panel at block 3108.

The verify lock mode subroutine checks to see if a lock mode was selected in decision block 3192. If a lock mode had not been selected then an error message is formatted in block 3196. When a lock mode is selected then a return constant is formatted in block 3194 and returned at return terminator 3198.

Typically, the dispatcher and/or supervisor will selected the process 20 when all the required data has been entered in the Get Code to Clear Covert Entry dialog box 18. Upon selecting the process button 20 the software will continue with decision block 3134 that verifies that the user is authorized to execute this function. If the user is not authorized then the program will return. When the user is authorized to obtain a clear covert entry combination the program flow will continue to block 3136 where the program obtains data from the display panel. Checking the authorization of a user is optional, but is preferred for security reasons.

Thereafter the program logic checks decision block 3138 to see if a lock name has been entered, if a lock name has not been entered, then an error message may be formatted in block 3140 and displayed in block 3128. After verifying that a lock name was entered in decision block 3138, the program flow may call the verify the lock mode subroutine 3190 discussed above. Thereafter, the software routine may verify that the lock mode was o.k. in decision block 3142. When the lock mode is o.k. the program logic continues to check the data to see if the lock was in the route mode in decision block 3144. When the lock is in the route mode the program logic checks to see if a route name was entered in text box 21 in block 3146. When route name is entered then it is preferred that the program logic verify that a route file exists in decision block 3148. If the route files does not exist an error message may be set up in block 3150.

When a route file does exist then the program logic preferably also verifies that a sequence number was entered in decision block 3152. When a sequence number was not entered an error message may be set up in block

3154. Thereafter the program logic moves to decision to 3156, which checks the return constant. If the return constant indicated an error then the lock program may move to check for the specific error that occurred and then format the appropriate error message in blocks 3167 through 3187 as discussed below.

When the return constant indicates no errors, the program logic continues to decision block 3158. This decision block checks to see if the lock is in the FLM or route modes so that the program logic may assign the appropriate route name and sequence number. For FLM mode locks this is accomplished in block 3160 and for route mode locks in block 3162. Thereafter the program logic may call the last combination subroutine 3164. This subroutine obtains the last combination for the lock from the lock database. Thereafter in decision block 3166 the program logic verifies that a combination was found. If the last combination was not found then the program logic would move to check for errors and format error messages in blocks 3167 through 3187 discussed below.

After finding the last combination program logic moves to checks to see if the lock record is open in decision block 3168. If the lock record is not open an error message may be formatted in block 3170 and the program logic moves to check for errors in decision block 3193 as discussed below. When the lock record is open the program logic moves to decision block 3172 where the program may check/test for valid lock data. If the data is invalid, then an error message is formatted in decision block 3174. When the data is valid, then the clear code is generated in block 3176.

The clear code combination is generated by applying a function to the combination issued to open the lock. This function may be any linear, non-linear or pseudo random function so long as a similar function is programmed into the lock. Consequently, the lock will recognize the modified combination as a covert entry clear combination. Preferably, the covert entry clear combination generator will use either a non-linear or pseudo random function so that it will be difficult and/or impossible for the lock user/operator to determine what the covert entry clear code combination would be from the issued combination.

After generating the clear code combination in block 3176 the program logic checks the lock mode in decision block 3178. This check is performed so that a proper covert entry code display window may be formatted as shown in either Figure 4A or 4B. If the lock is a route lock the message is formatted in block 3182 and thereafter the program logic checks to see if the lock is currently part of a route in decision block 3184. If the lock is on a route then the route sequence number message would be formatted in block 3186.

If the lock mode is a FLM mode lock then the appropriate message would be formatted in block 3180 thereafter the program logic moves to format the code to be utilized to clear the covert entry in blocks 3188 and 3189. Thereafter the display message with the clear code could be displayed in 3191. Exemplary clear code message boxes 22A and 22B are shown in Figures 4A and 4B. Thereafter an activity log record could be generated in block 3197 and the program logic would return to the Get Code to Clear Covert Entry display panel 3108 after the user clicks or selects the o.k. button 24.

Figures 31A-1 and 31B-1 illustrate the program logic that could be utilized to permit a user and/or supervisor to dispatch a lock (issue an opening combination) from the Get Code to Clear Covert Entry dialog box 18 shown in Figure 3B. Consequently, a user and/or supervisor may obtain a covert entry reset combination without first dispatching a service call to the lock.

A Dispatch Svc Call check box 27 was added to the Get Code to Clear Covert Entry dialog box 18 shown in Figure 3A resulting in the dialog box shown in Figure 3B. Figures 31A-1 and 31B-1 are modifications of Figures 31A and B. These modifications enable the user and/or supervisor may obtain a covert entry reset combination without first dispatching a service call to the lock.

A decision block 3111 that updates check box 27 is the only modification shown on Figure 31A-1. Blocks 3151, 3153, 3155, 3157, and 3159 add the ability to obtain a covert entry clear code without having to first dispatch the lock. When the user selects/checks the dispatch check box 27 the program logic, in decision block 3151, directs the program flow to the dispatch a service call subroutine 5900. Preferably, for security reasons, the logic prevents calling this subroutine if the lock selected is in the route mode and currently assigned to an active route (a route with more that one lock) with decision block 3153. In order to inform the user of this condition an error message may be formatted in blocks 3153 and 3155. Additionally, it is desirable to avoid processing locks with errors by employing decision block 3159.

Figures 59, 60 ,61A, and 61B provide examples of the modifications to the dispatch a service call subroutine. This routine generates the combination to be entered to open a particular lock. This routine was modified so that the

lock parameters entered in the Get Code to Clear Covert Entry dialog box 18 would be sufficient to obtain a combination without affecting the normal operation of this subroutine in dispatching locks in other modes.

Figure 59 illustrates the changes to the dispatch a service call subroutine 5900. Preferably, the lock name box is disabled when dispatching a service call to a lock with a covert entry. Blocks 5904, 5906, and 5908 provide for disabling the lock name box. As a result of this feature the Dispatch a FLM Service Call dialog box 25 shown in Figure 3C is displayed. This feature also prevents the covert entry reset combination from being issued for a different lock. When the process button 29 is selected or clicked the dispatch service subroutine 6000 is called. Thereafter, if the subroutine 5900 was called as a result of a covert entry the subroutine returns the opening combination to the get covert entry clear code subroutine 3100 to modify the opening combination into a covert entry reset or reset and open combination at blocks 5910 and 5912.

Figure 60 illustrates the program logic for the dispatch service subroutine 6000 called from Figure 59. The changes to this subroutine are included in the changes to the generate combo subroutine 6100. Preferably, the dialog box is closed prior to the subroutine returning to subroutine 5900 with blocks 6002 and 6004.

Figures 61A and B illustrate the generate combo subroutine 6100 that is called from Figure 60. Preferably, this subroutine is modified by the addition of decision block 6102. This decision block prevent the opening combination from being displayed after it is generated. Consequently, security of the system is

increased since the user would not see both the opening combination and covert reset combination on the screen at the same time.

Figures 32A and 32B illustrate the program logic used to list the activity log shown in Figure 14. This activity log could be generated after the user selected the List Activity Log button 62 on the Supervisor Reports Menu 60 shown in Figure 13. The changes to this subroutine are illustrated in decision block 3202, which adds a check for a clear covert entry event and if one is found formatting the appropriate display line for clearing a covert entry in block 3204. An exemplary line entry is shown in Figure 14 at reference numeral 70. This subroutine begins at start terminator 3200 and is called from the get log data subroutine 3600 shown in Figure 36.

When a user typically a supervisor wishes to display an activity log they would select the supervisor reports menu from the supervisor menu 14 shown in Figure 2 by selected the reports menu button 15. Changes to the activity log routine 3400 backup activity log routine also 3400 and the display contents of SA subroutine 3306 were made to accommodate the covert entry feature. When the user selects either the List Activity Log button 62 or the List a Backup Log File button 63 the initialization dialog box subroutine 3400 would be initialized. This subroutine is illustrated in Figures 34A, B and C. The changes to this subroutine are encompassed in the display activity log report subroutine 3500 calls to which are made in Figure 34B and Figure 34C.

The display activity log report subroutine 3500 is illustrated in Figure 35. The changes to this subroutine are shown in changes to the get data for an access group subroutine 3600 is called twice in this subroutine. Figure 36

illustrates the get log data subroutine called from Figure 35. The changes to this subroutine where discussed above in the list audit subroutine 3200.

Cover Covert Entry Closed Code

The employment of an altered/modified close seal enables the lock dispatcher to be promptly notified when the back cover or other portion of a lock is tampered with while the lock was open. Figure 27 illustrates the flow chart index to assist one of ordinary skill in the art of understanding the following flow charts.

Figure 28 illustrates the program logic utilized to check the close seal for a cover covert entry occurring during a service call. This program logic adds a generate cover covert entry close seal subroutine 4600. After generating the cover close seal the program compares the modified or cover covert close seal with the close seal provided by the lock and/or by the operator in the field in decision block 3840. If the covert entry closed seal matches the actual closed seal called in or provided on an electronic key, then the service call is closed with an indication that a cover covert entry occurred during the call. If the closed seal still does not match then an error is indicated.

A service call may be closed in one of three ways. A service call may be closed using the FLM services menu 26 by selecting the Closed Service Call button 28. Alternatively, the service call may be closed using the Route Services Menu 30. On the Route Service Menu, a service call may be closed by utilizing the Close Service Call button 32, the Close a Route Using Keys button 31, or Close a Route by Using Menu button 33.

The close a service call subroutine 3900 is accessed by using either the Close Service Call button 28 on the FLM Services Menu 26 or the Close Service Call button 32 on the Route Service Menu 30. The changes to the close a service call subroutine 3900 are contained in the close lock subroutine 4000 discussed below.

Figure 40 illustrates the close lock subroutine 4000. The changes to this subroutine are included on the verify the close code subroutine 4100 discussed below and the verification that a cover covert entry did not occur in decision block 4002. If a cover covert entry occurred then a message would be formatted in block 4006 which indicates that the lock was closed and that a cover covert entry event occurred. An exemplary dialog box showing this is shown in Figure 8 at dialog box 40. If on the other hand a covert entry event did not occur then a message would be formatted indicating the lock was closed in block 4004.

The verify closed code subroutine 4100 is illustrated in Figures 41A, B and C. The changes required for the covert entry feature is shown in Figure 47B after the program logic finds that the expected close seal does not match the entered close seal. In this case the cover covert entry close seal subroutine 4600 is called. Thereafter, in decision block 4106 the program logic checks to see if the covert entry close seal generated matches the entered close seal. If the close seal does not match then the program flow returns to increment the attempts to close the lock at block 4107. If the generated covert entry close seal matches the entered close seal then a cover covert entry message is stored in the audit record extension in block 4108.

The Close a Service Call dialog box 34 shown in Figure 7 is an exemplary dialog box that may be displayed. The dispatcher/operator would enter the ATM/lock name in text box 35 and the close seal in text box 36. Upon selecting or clicking the process button 38 the program would compare the entered close seal against an expected close seal or a cover covert entry closed seal as discussed above. When a cover covert entry event occurred based on the close seal entered, the dialog box display could be the close a route and/or services call or close a FLM services call message box 40 shown in Figure 8.

Figures 42A, B and C display the program logic utilized when the user selects the close a route using menu button 33 on route services menu 30. The close a route using menu subroutine 4200 will display the close a route using menu dialog box 44 shown in Figure 9. This dialog box permits the user to enter the route name in text box 41 the route service ID (user ID of the individual opening the lock) in text box 43. If the lock was a dual mode lock the second route service ID in text box 45. The changes to this subroutine include changes to the close route subroutine 4400 and the display messages for each lock with error subroutine 4300. Additionally, decision box 4204 and box and format message blocks 4206 and 4208 are added. Preferably, these functions are added after the program has checked for errors and found no errors in decision block 4202. Thereafter in decision block 4204 the program logic checks to see if a covert entry message or other error message was not empty. If the message was empty then the standard locks closed message would be generated in box 4208. However, if the error message was not empty then a message indicating that the cover covert entry occurred and that the

locks were closed would be generated in block 4206. An exemplary dialog box showing this message is shown in Figure 12 Enter Close Code for Route dialog box 56.

The close a route use a menu subroutine 4200 may display the enter Close Codes for Route dialog box 48 upon the user selecting the process button 46 on the Close a Route Using Menu dialog box 44. The Enter Close Codes for Route dialog box 48 has a text box 50 which permits the user to enter close codes for each lock in the route.

Figures 43A and B illustrate changes to the display messages for locks with errors subroutine 4300. The additions to this subroutine are decision block 4304 where the program checks for a cover covert entry close code. If a cover covert entry close code was found then a cover covert entry message would be formatted in block 4306. Preferably, this function would be inserted after checking for errors in decision block 4302.

The close route subroutine 4400 is illustrated in Figures 44A, B and C. These changes include the addition of block 4402, which initializes the variables used for the cover covert entry feature when starting the subroutine. Additional changes include storing the covert entry variable(s) at the end of the routine in block 4420 shown in Figure 44C. When the expected close seal does match the close seal entered in decision block 4409, additional covert entry function specific variables are set to defaults in block 4410. Next the cover covert entry close seal is generated using subroutine 4600. If the generated covert entry close seal matches the entered close seal in decision block 4412, then a variable indicating the covert entry is set to in block 4414. Thereafter the program flow would return to the find the log record for the

current lock in subroutine 4416. It is preferable that these additions be placed at a convenient in the program and prior to updating any lock records when the closed seal entered and the generated close seal do no match.

Figures 45A, B and C illustrate the program logic used to close a route using electronic keys. This program logic could be accessed by selecting the Close a Route Using Keys button 31 on Route Services Menu 30 shown in Figure 6. The changes to this program logic are similar to those discussed above for the programming logic used to close a route using menus. The changes to this subroutine are included in changes to the display error message subroutine 4300 discussed above and the close route subroutine 4400 also discussed above. Similarly blocks 4512, 4514 and 4516 have been added to display the appropriate messages. Block 4516 formats the message that all the locks were closed and block 4514 will format the message to indicate that all locks were closed and that a cover covert entry occurred. Figure 11 illustrates an exemplary Close a Route Using Keys dialog box 52. Upon selecting the Process button 54 the computer program would read the electronic key in the key reader attached to the computer to obtain the close seal from each lock in the route.

Figure 46 illustrates the check for cover covert entry close code subroutine 4600. This subroutine compares the entered closed seal with an expected cover covert close seal that should have generated by the lock if the back cover of that lock was removed. Thus, this subroutine may detect lock tampering by detecting removal of the lock's back cover while the lock was open.

This subroutine begins at start terminator 4602 and then initialized the return constant to a pre-selected number, in this case 27, in block 4604. Thereafter the expected close seal and other lock data is loaded in block 4606. Since this data is loaded as an ascii string, this string is converted to hexadecimal in block 4608 for future processing. This conversion is verified in decision block 4610. When the conversion was error free the program flow moves to generate the cover covert close seal in block 4612.

This subroutine employs the same function as utilized in the lock when the lock generates a cover covert entry close seal by modifying a normal closed seal. The function utilized can be a linear, non-linear and/or pseudo random function. A non-linear or pseudo random function could be utilized so that the lock operator would be unable to determine or would have a very difficult time in reporting a close seal that would indicate the lock was properly closed when the operator had in fact tampered with the lock by removing the back cover.

Thereafter the program would check in decision block 4614 to see if the cover covert close seal matched the original (expected) close seal. If there is a match then the cover covert close seal would be altered in a known manner in block 4616. Next the cover covert close seal may be converted to ascii in block 4618. This close seal would be then compared to the entered close seal in decision block 4620. If these close seals did not match then the program would at return terminator 4624. When there was a match, indicating that a back cover covert entry occurred, then a message would be formatted indicating such in block 4622.

Utilizing both the get covert entry combination and the close seal comparison routines discussed above an dispatcher and/or supervisor would

be able to both clear a covert entry event (covert entry, attempted covert entry and/or tampering), and to be promptly notified of a back cover covert entry event which would indicate potential for theft and/or lock tampering by an insider.

Lock Software/Firmware

The flow charts for the lock software are illustrated in figures 47 through 58.

Only the additions/changes to software/firmware are discussed in detail.

Figures 47 through 58 provide the changes made to the firmware/software that is in the lock shown in pseudo code format. The changes to the lock firmware/software enable the locks microprocessor to periodically to check for a covert entry event with sufficient periodicity to detect an covert entry event. The two types of covert entry detected are back cover removal (one indication of tampering), which may be sensed with a switch other device that would cause a change of state that could be recognized by the microprocessor or electronic control and bolt movement without an authorized combination entry. The software/firmware may also provide a close seal/code that indicates that the lock was properly closed or if the back cover was removed while the lock was open. Additionally, the software/firmware provides the ability to reset or clear a covert entry event so that the lock may be opened by a person with an authorized combination. The combination or code utilized to reset/clear the "lock out" is a combination or code that can not be utilized twice in a row.

Figures 47A and B illustrate in pseudo code form the inserts into the main line assembly program. Periodically, throughout the main line program a

call is made to the check for covert bolt movement subroutine 4800. A call to this subroutine must periodically be made during normal microprocessor operation to ensure that a covert event (back cover removal or bolt movement without entry of a valid combination) is both monitored and detected during the covert event. Other changes to the main line program are principally done for housekeeping purposes or to set up the main line program so that the subroutines start from a known state. These changes are anticipated to be understandable by one of ordinary skill in the art of programming and will not be discussed in detail since the pseudo code provides sufficient information.

Additionally, the main line program makes calls to other subroutines that have been altered to enhance the lock performance and to enable covert entry detection or resetting the lock after a covert entry event. These subroutines include the subroutine for saving the current flags to an output record buffer 4900 and a subroutine for removing flags from an input record buffer to RAM 5000. Additionally, the calls to change key in subroutine 5800 and wait for touch memory subroutine 5700 were altered by the addition of the check for covert bolt movement subroutine 4800 as is discussed below. Additionally, the combo in subroutine in 5100 has been altered to check for covert bolt movement as well as to check for a covert entry reset combination. The main line subroutine also makes a call to the check for open subroutine 5200 that was altered to test for tampering.

Figures 48A and B provide exemplary pseudo code for the check bolt covert subroutine 4800. This subroutine checks for either the back cover removal or bolt movement. This subroutine will also check to see if the lock has been opened by a combination entry. When the lock has been opened by

a combination entry then normal lock operation would continue if the lock was not opened using a combination entry then a covert entry flag would be set. Additionally, if the back cover was removed a covert entry flag is set along with a back cover removed flag. The use of the back cover removed flag enables the lock to differentiate from a covert entry caused by bolt movement without a combination entry and a covert entry event caused by removing the back cover.

After the check bolt covert subroutine 4800 is called the program checks for either bolt movement or back cover removal (the bolt power supply flag is checked). If either event occurs, then the subroutine continues. When the back cover is in place and the bolt is not moving (the bolts power supply is off or the flag not set), then the subroutine will return. Next, the watch dog timer is reset, while not required this helps the lock program continue to function smoothly.

Additionally, when the bolt power supply is on this indicates that power to the microprocessor is being provided by a second power supply which is typically only used for powering a microprocessor when a covert entry event has occurred without the lock being powered. Thus, the lock is powered up if the bolt is moved.

Resetting the watch dog timer ensures that the program will function properly when the microprocessor is "Woken up" by a bolt movement. Furthermore, the subroutine may check for the housekeeping records in memory (RAM). If these records are not located in RAM then the subroutine would read these records into RAM and set a flag which indicates that the records are present. The subroutine checks to see if the back cover was removed by checking the back cover switch. It is desirable for any covert entry

to clear the last users ID number, if utilized, and the time stamp on the audit record since some current commercial embodiments of the lock do not have a clock and the actual identity of the person making the covert entry may be unknown. Additionally, a covert entry flag is set together with the back cover removed flag. The setting of both the covert entry flag and the back cover removed flag enables the lock to discern what type of covert entry occurred, i.e., a back cover covert entry or a bolt movement covert entry. Preferably, these audit records indicating the covert entry and the flag status are written into non-volatile memory.

Thereafter the power supply may be latched. It is preferable to latch the power supply to ensure for a period of time sufficient to write the housekeeping records and to shut down the microprocessor. When the back cover switch indicates that the back cover has not been removed, the subroutine checks for the direction of bolt movement while the bolt power supply in on. This may be accomplished by checking the bolt extended flag when the bolt is extended this flag is set to 0 and therefore bolt motion would be in the opening direction and so the program sets both the bolt direction flag to opening and the bolt open flag to open. Alternatively, if the bolt direction is closing then the bolt direction flag may be set to closing. Thereafter, the subroutine checks to see if the lock was opened by combination entry.

When the lock was opened by combination entry then the bolt flags will be saved to non-volatile memory to maintain audit records. When the lock was not opened by combination entry then the covert entry flag or other indicator could be set. Preferably, the user information in the lock would be cleared prior to making an audit record. When audit records are maintained it is useful to

check for an audit record write error, in which case the lock firmware/software could be restarted. Thereafter, in the preferred embodiment, the lock power is latched and flags to set to check for a full dial turn to ensure that the lock is closed. If a full turn is detected then the bolt may be marked as closed using the bolt closed flag and that flag would be saved to non-volatile memory. Alternatively, if a full dial turn was not detected the lock could wait until the bolt power supply turns off and stays off for a pre-specified period of time, approximately 25 milliseconds, or the bolt was opening that is now closing for 25 milliseconds, or the bolt was but is now opening for 25 milliseconds. Thereafter, it is preferred that the processor restart flags are cleared and the processor checks the voltage of the capacitor. If the capacitor voltage and normal power supply capacitor is not sufficiently high then the power is unlatched. Thereafter the capacitor is drained. If the voltage to the microprocessor is sufficiently high the knock off release may be enable and the knock off operation performed thereafter the knock off release may be disabled and the lock firmware will be restarted.

Figures 49 and 50 provide pseudo code showing the changes to the non-volatile memory saved routine 4900 and the non-volatile memory read routine 5000. The additions to these subroutines provide the ability to save the covert entry flags to the output record buffer or to read the covert entry flags from the input record buffer to RAM respectively.

Figures 51A and B illustrate the combo in subroutine 5100. The principle changes to this subroutine include changes to the generate final subroutine 5500 several periodic calls to the check for covert bolt movement subroutine 4800 and changes to the general delay subroutine 5400. As

discussed above the periodic calls to the covert bolt movement subroutine 4800 are performed to ensure that a covert entry event is not missed during microprocessor operation. The combo in subroutine 5100 is called from the main line program 4700.

Figures 52A and B provide the exemplary pseudo code for the check for open subroutine 5200 called from the main line program 4700. The check for open subroutine 5200 is placed in the main line program at a point where it is expected that the lock should be closed because the lock was just powered up. Consequently, if the lock is indicated open then the lock will set a covert entry flag and audit record.

The lock appearing to be open when the lock was actually closed would be an indication that lock was tampered with. Thus, this subroutine will check to see if the lock was opened in either the factory mode or in the FLM, route or bank modes. If the lock appeared to be opened in the factory made the program would further check to see if the bolt was extended or the bolt was retracted with the last bolt movement being in the closed direction. In this case, the factory mode open would be cleared and the bolt open flag would be cleared and the flags set to the non-volatile memory. However, if the lock indication was open in the factory mode and the bolt was retracted, a covert entry flag would be set and a covert entry recorded in non-volatile memory.

If the lock indication was open in either the FLM, route or bank modes and if the bolt was indicated as being open, then a covert entry audit record and covert entry flag would be set. After setting the covert entry flag the subroutine could return after writing the appropriate audit record. When the lock indication was open and the bolt was in the extended position then the

subroutine checks to see if the back cover was removed. If the back cover was removed, then the close seal is modified using the back cover seal subroutine 5300. Thereafter the modified closed seal would be placed in the audit record. The modified closed seal would be utilized to indicate to the dispatcher and/or supervisor that the lock had been tampered with by removing the back cover while the lock was open.

Figure 53 shows the back cover seal subroutine 5300 in pseudo code form. This subroutine begins by placing appropriate values into a data buffer. These values may be combined using a function. Preferably, this function would be a pseudo random function. The use of a pseudo random function would make it very difficult to impossible for the lock user to be able to predict and determine whether the closed seal was modified or not. If the modified closed seal is the same as the old close seal then the modified close seal would be changed to ensure that it is different from the expected close seal and thus enable the dispatcher and/or supervisor to determine that the back cover had been removed while the lock was opened.

Figure 54 illustrates the changes to the general delay subroutine 5400. These changes include adding the check for covert bolt movement subroutine 4800. The addition of this subroutine ensures detection of a covert entry. The general delay subroutine 5400 is called from the combo in subroutine 5100.

The generate final subroutine 5500 is called from the combo in subroutine 5100. Figure 55 illustrates the pseudo code that was inserted into the generate final subroutine 5500 to add the covert entry functionality. This pseudo code checks if a covert entry has been detected. When a covert entry is detected and the operator opening the lock is not a bank mode user then the

subroutine will generate a new combination from the expected authorized combination value. This new combination would be the covert entry reset combination. In the preferred embodiment the reset combination could act as both a covert entry reset and lock opening combination. The function utilized to covert the old combination into the reset combination could be a linear, non-linear or pseudo random function. Preferably, the function would a pseudo random function to make it more difficult for the person opening the lock to determine what the covert entry clear combination from an authorized opening combination.

Figure 56 illustrates the save open subroutine 5600 with the additions shown in pseudo code. This change entails a flag indicating that the bolt was not closed when a combination was entered.

Figure 57 illustrates the changes in pseudo code to the wait for touch memory subroutine 5700 that is called from the main line program 4700. The addition to this subroutine is the call for the covert bolt movement subroutine 4800 as discussed in detail above.

Figure 58 illustrates the change key in subroutine 5800 that is also called from the main line program 4700. The insert into this subroutine sets up the flags in non-volatile memory for restarting the lock and processing the check for open subroutine 5200 upon removing the change key and restarting the lock. These changes complement changes in the check for open subroutine 5200 performed to enable the covert entry functionality.

In summary, numerous benefits have been described which result from employing the concepts of the invention. The foregoing description of a preferred embodiment of the invention has been presented for purposes of

illustration and description. It is not intended to be exhaustive or to limit the invention to the precise form disclosed. Obvious modifications or variations are possible in light of the above teachings. The embodiment was chosen and described in order to best illustrate the principles of the invention and its practical application to thereby enable one of ordinary skill in the art to best utilize the invention in various embodiments and with various modifications as are suited to the particular use contemplated. It is intended that the scope of the invention be defined by the claims appended hereto.

CLAIMS

We Claim:

1. A method for resetting the covert entry feature of electronic combination lock:

entering a one-time covert entry resent combination into said electronic combination lock;

comparing said entered covert entry reset combination with an authorized covert entry reset combination;

after said entered covert entry reset combination compares equal with said authorized covert entry combination, resetting said covert entry feature of said electronic combination lock, whereby the entry of an authorized opening combination will enable said lock to be opened.

2. A method for resetting the covert entry feature of electronic combination lock:

generating a one-time covert entry reset combination;

transmitting said covert entry reset combination to an operator;

entering said covert entry resent combination into said electronic combination lock;

comparing said entered covert entry reset combination with an authorized covert entry reset combination;

after said entered covert entry reset combination compares equal with said authorized covert entry combination, resetting said covert entry feature of said electronic combination lock, whereby the entry of an authorized opening combination will enable said lock to be open.

3. A method for resetting the covert entry feature of electronic combination lock:

obtaining a one-time covert entry reset combination;

entering said covert entry reset combination into said electronic combination lock;

comparing said entered covert entry reset combination with an authorized covert entry reset combination;

comparing said it entered covert entry reset combination with an authorized opening combination;

after said entered covert entry reset combination compares equal with said authorized covert entry reset combination, resetting said covert entry feature of said electronic combination lock;

after said entered covert entry reset combination compares equal with said authorized opening combination and after clearing said covert entry feature, enabling said lock to be opened.

4. A method for resetting the covert entry feature of electronic combination lock:

generating a one-time covert entry reset combination;

transmitting said covert entry reset combination to an operator authorized to reset said covert entry feature of said lock;

entering said covert entry reset combination into said electronic combination lock;

comparing said entered covert entry reset combination with an authorized covert entry reset combination;

comparing said it entered covert entry reset combination with an authorized opening combination;

after said entered covert entry reset combination compares equal with said authorized covert entry combination, resetting said covert entry feature of said electronic combination lock;

a after said entered covert entry reset combination compares equal with said authorized opening combination and after clearing said covert entry feature, enabling said lock to be opened.

5. A method for resetting the covert entry feature of electronic combination lock:

obtaining a one-time covert entry reset and open combination;

entering said covert entry reset and open combination into said electronic combination lock;

comparing said entered covert entry reset combination with an authorized covert entry reset and open combination;

after said entered covert entry reset and open combination compares equal with said authorized covert entry reset and open combination, resetting said covert entry feature of said electronic combination lock;

after said entered covert entry reset and open combination compares equal with said authorized covert entry reset and open combination, enabling said lock to be opened.

6. A method for resetting the covert entry feature of electronic combination lock:

generating a one-time covert entry reset and open combination;

transmitting said covert entry reset and open combination to an operator;

entering said covert entry reset and open combination into said electronic combination lock;

comparing said entered covert entry reset and open combination with an authorized covert entry reset and open combination;

after said entered covert entry reset and open combination compares equal with said authorized covert entry reset and open combination, resetting said covert entry feature of said electronic combination lock;

after said entered covert entry reset and open combination compares equal with said authorized reset and open combination, enabling said lock to be opened.

7. A computer program that generates a covert entry reset combination, said combination is not capable of immediate reuse, whereby said combination will not reset a covert entry twice in a row.

8. A computer program that can detect the removal of a lock's back cover when a bolt of said lock is retracted by the close seal output from the lock.

9. The invention disclosed herein.

Main Menu

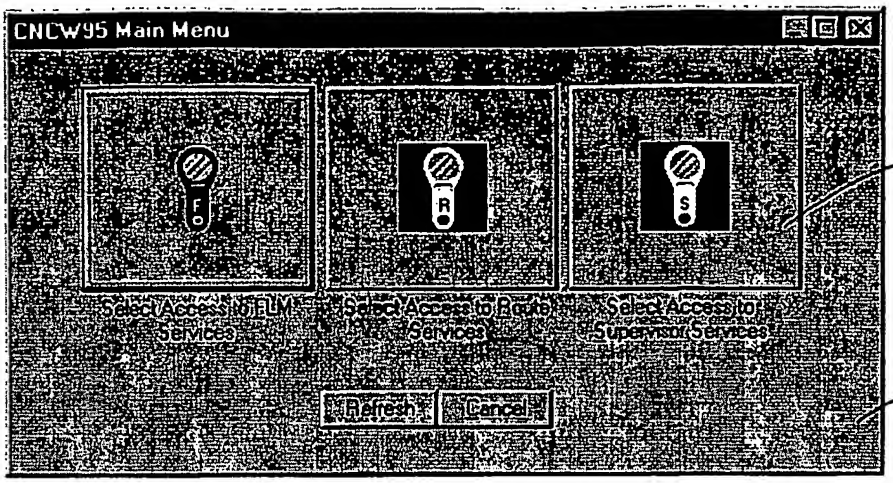


FIG. 1

Supervisor Services Menu (Accessed from the Main Menu)

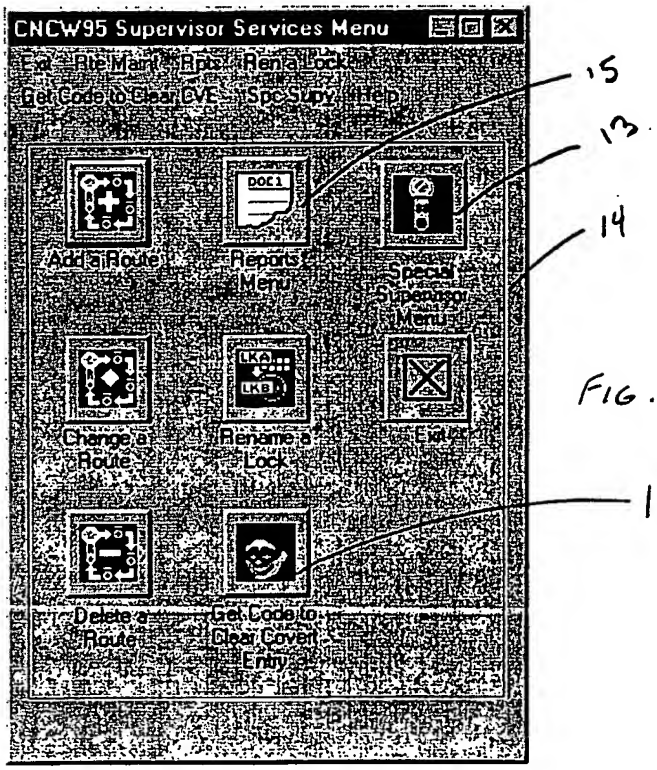


FIG. 2

Covert Entry Clear Code Menu (Accessed from the Supervisor Services Menu)

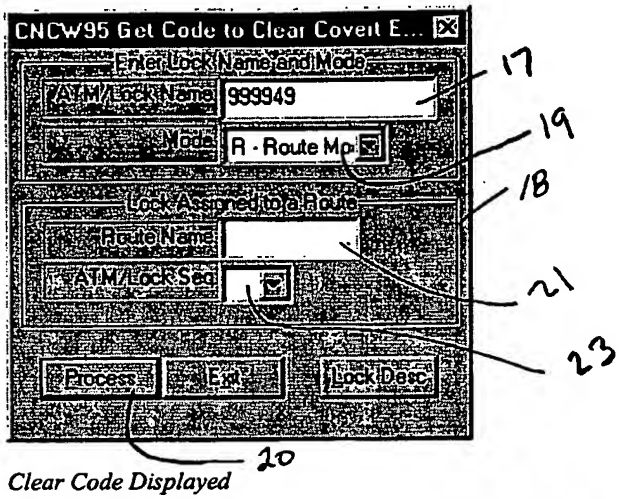


FIG. 3A

Clear Code Displayed

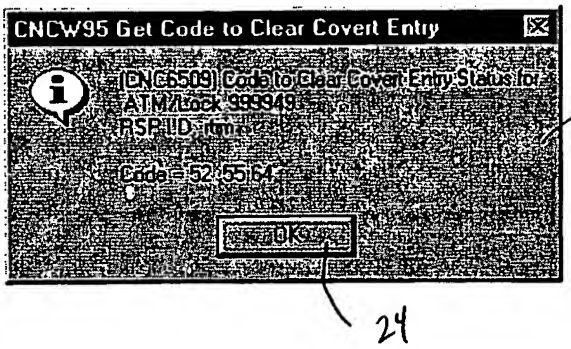


FIG. 4A

Covert Entry Clear Code Menu (Accessed from the Supervisor Services Menu)

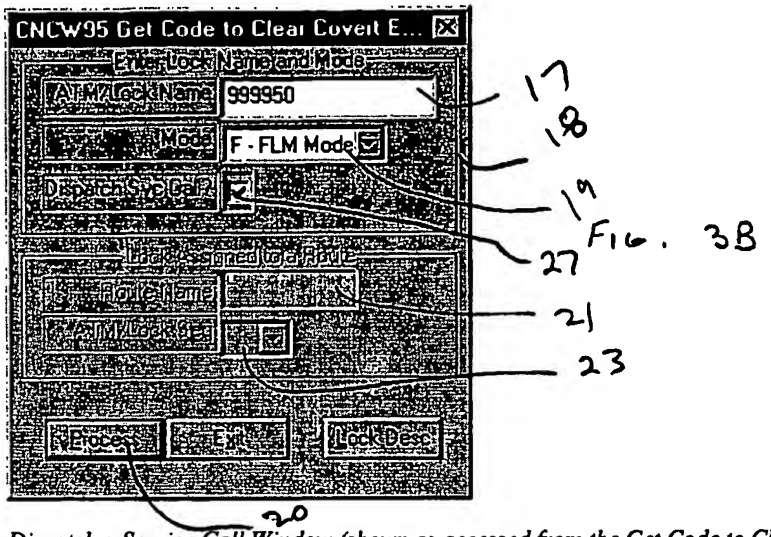


FIG. 3B

Dispatch a Service Call Window (shown as accessed from the Get Code to Clear Covert Entry window) – This window will only show if the Dispatch Service Call box is checked.

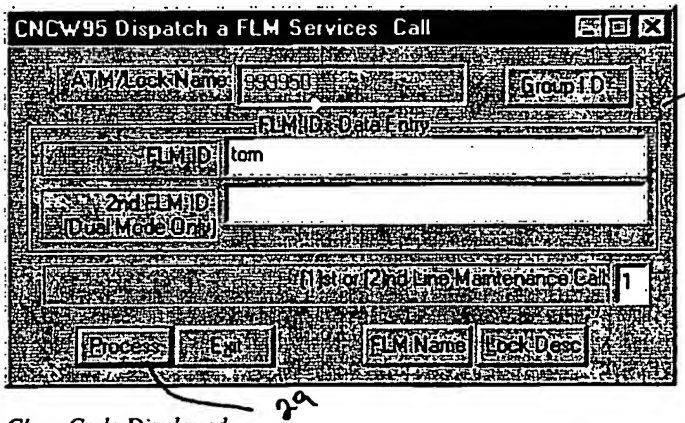


FIG. 3C

Clear Code Displayed

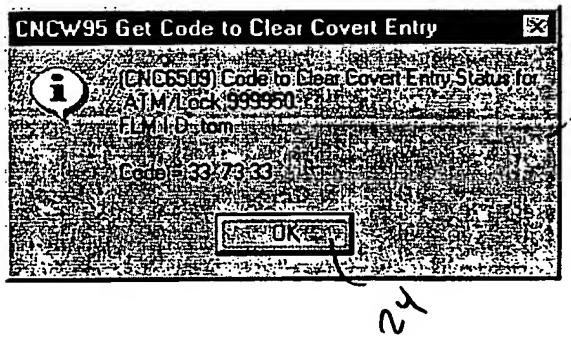


FIG. 4B

FLM Services Menu and Route Services Menu (Accessed from the Main Menu)

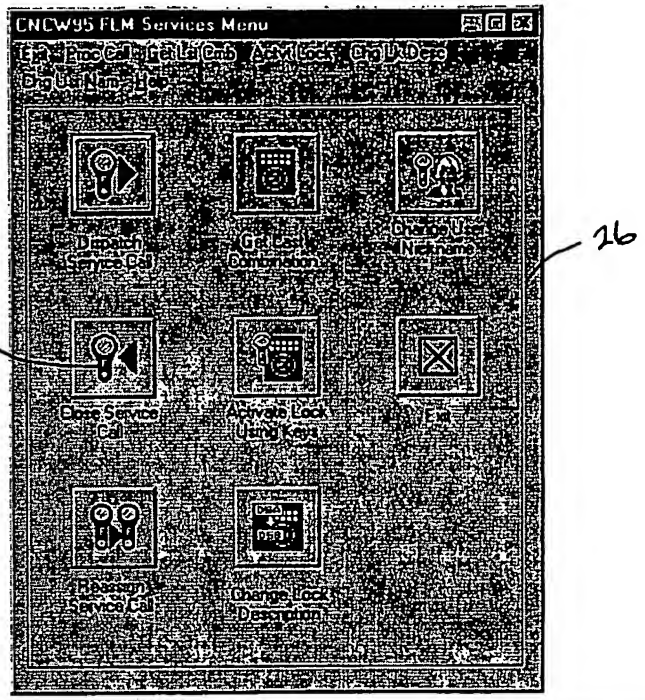


FIG. 5

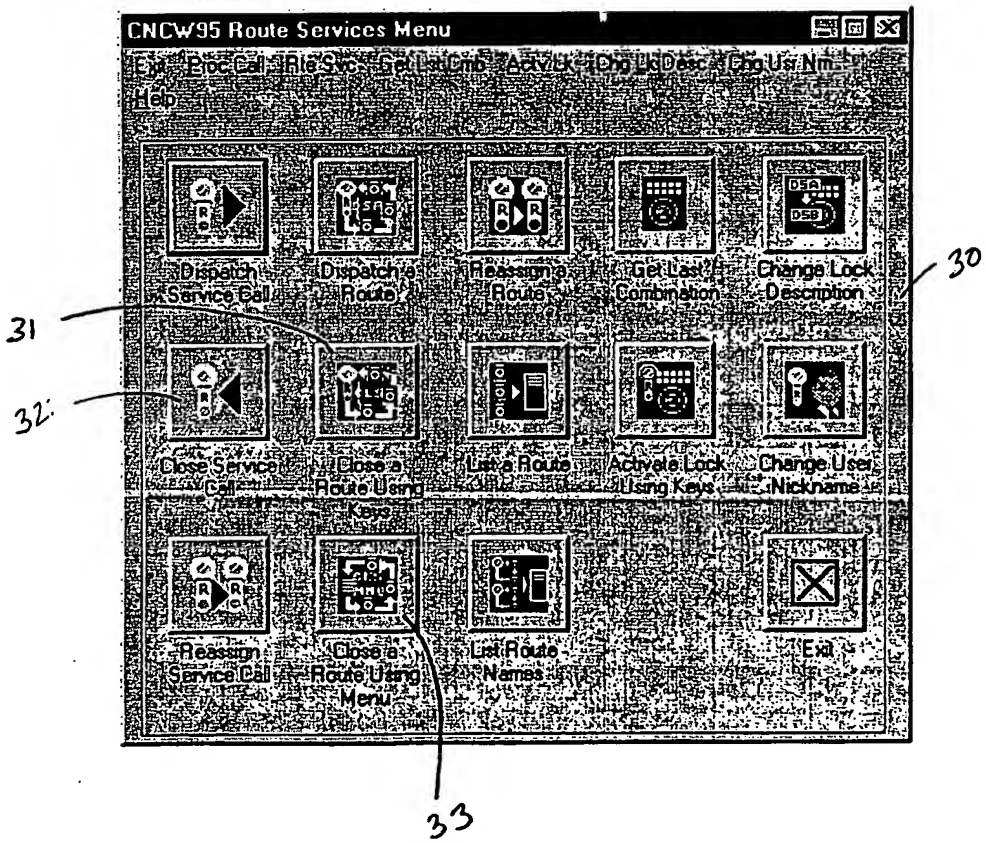


FIG. 6

Window to Close a Service Call (Accessed from the FLM or Route Services Menus)

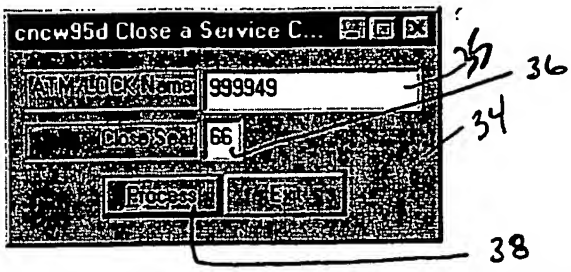


FIG. 7

Message displayed when a cover covert entry occurred (when the call is closed)

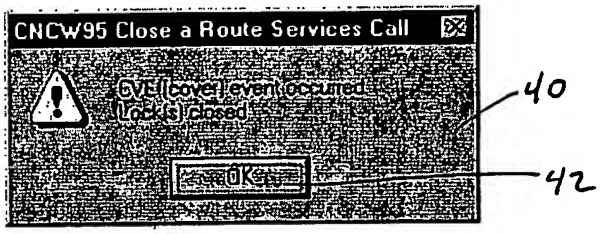


FIG. 8

Close a Route Using Menu and Close a Route Using Keys Menus (Accessed from the Route Services Menu)

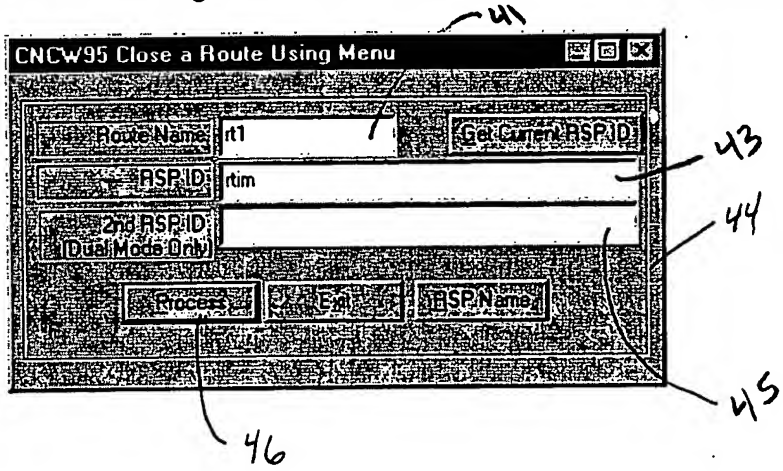


FIG. 9

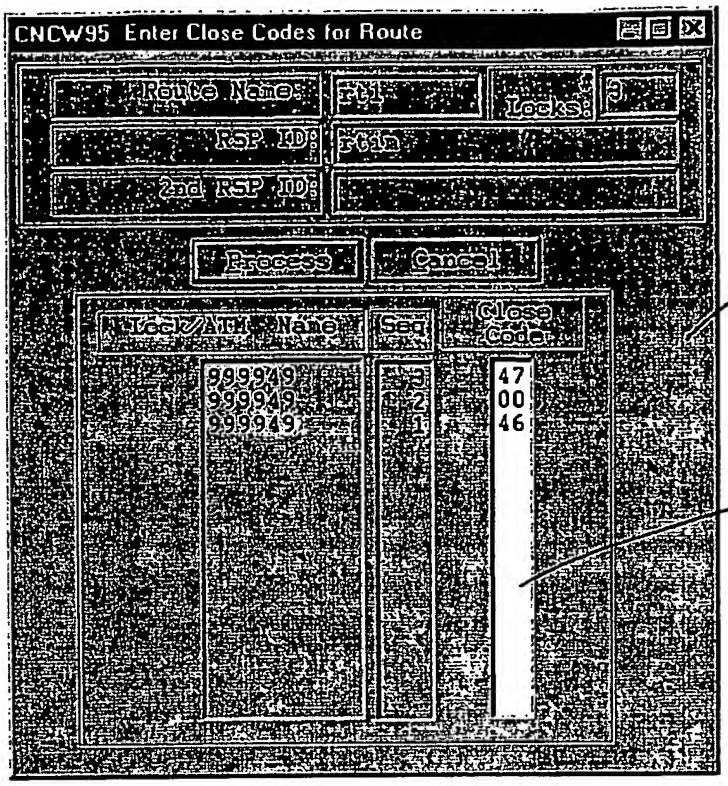


FIG. 10

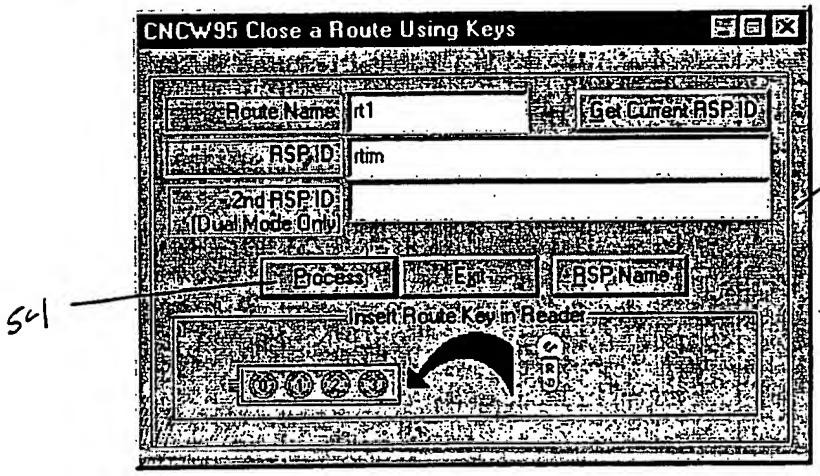


FIG. 11

Message displayed when a cover covert entry occurred (when the route is closed)

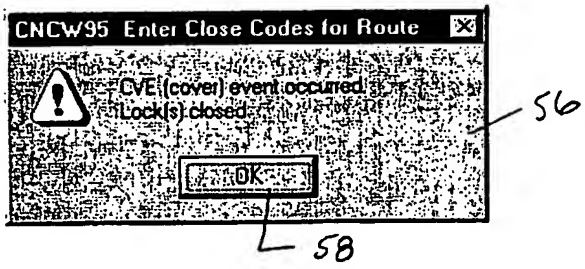


FIG. 12

Reports Menu (Accessed from the Supervisor Services Menu)

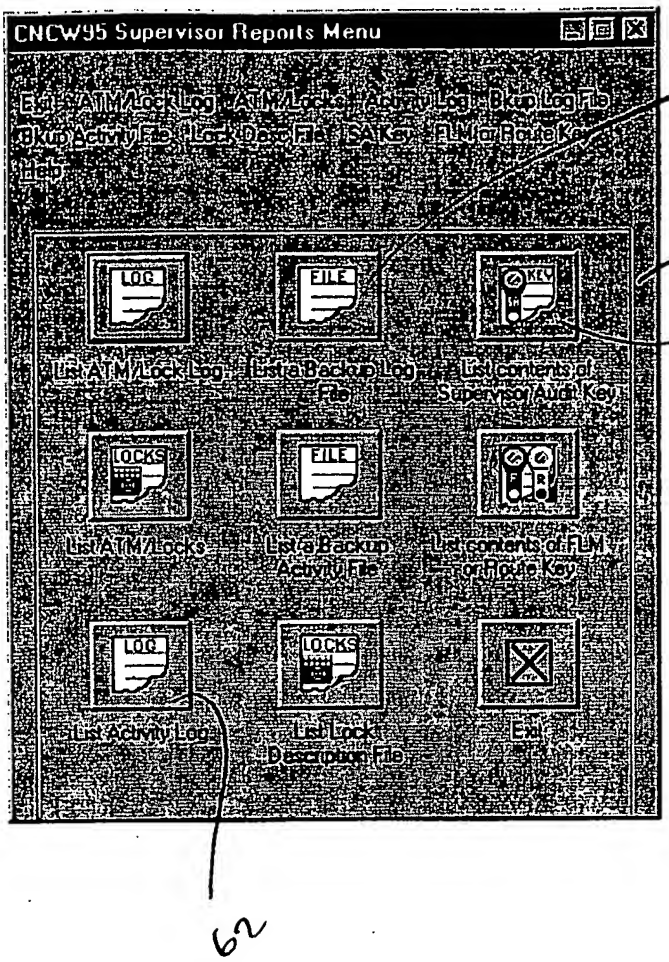


FIG. 13

Activity Log Report showing the following events (Accessed from the Reports Menu):

1. *A cover covert entry on a close service call*
2. *The issuing of a covert entry clear code*
3. *A cover covert entry on a close route*

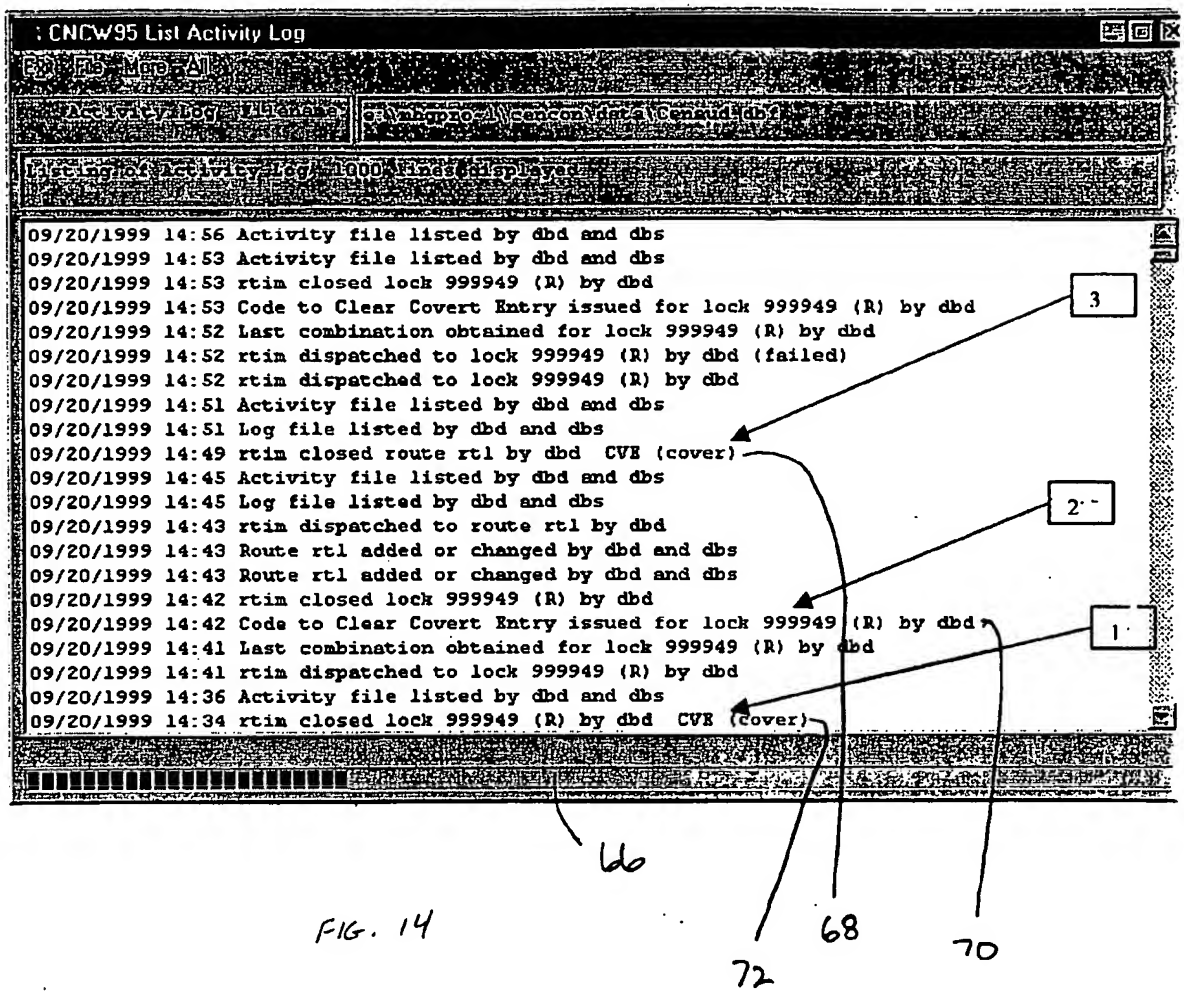


FIG. 14

List Contents of Supervisor Audit Key report showing the following events in a lock audit download (Accessed from the Reports Menu):

1. *A covert entry*
2. *A cover covert entry*
3. *A clear code entry after a covert entry*

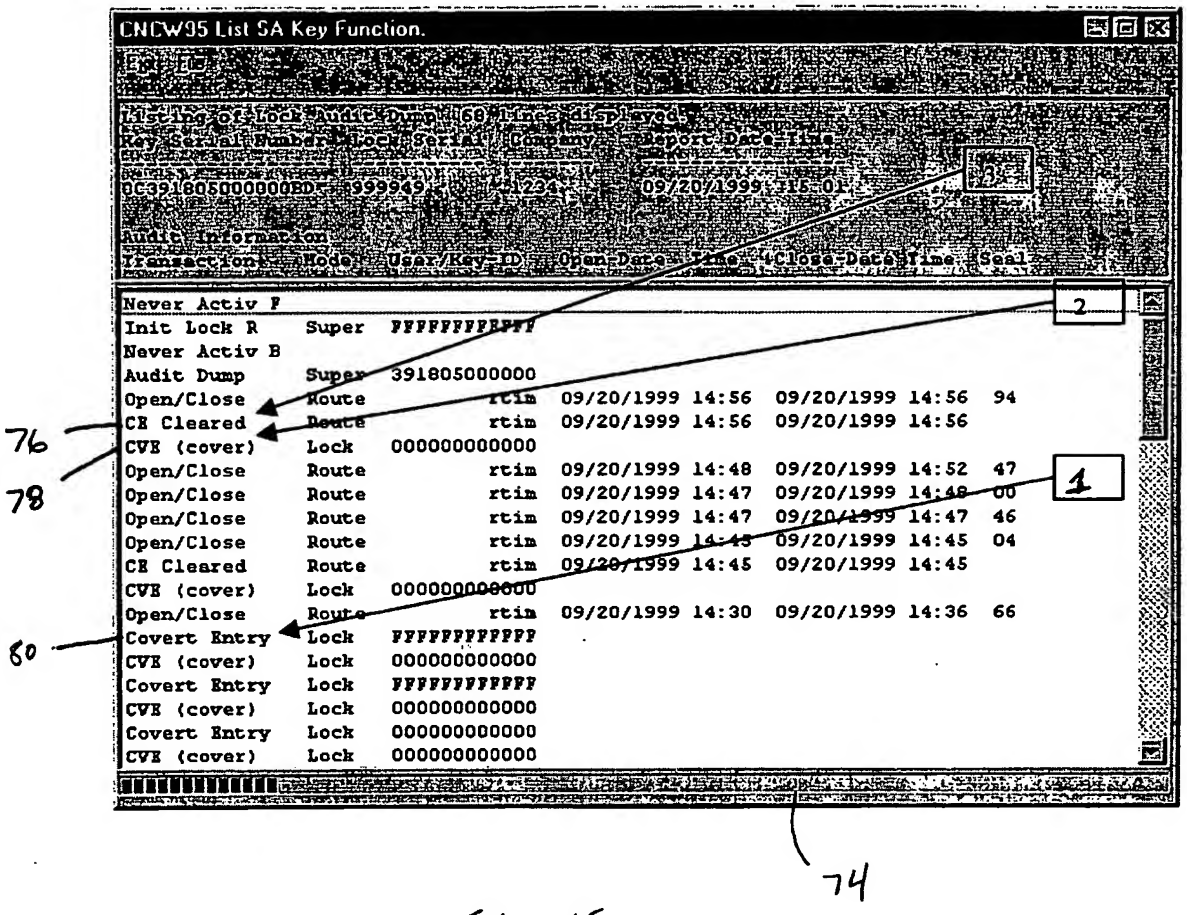


FIG. 15

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature

Introduction

The Covert Entry Feature causes an ATM/Lock to respond to tampering by recording the incident and terminating operations until a special 'Clear Covert Entry Code' is entered. The next time the ATM/Lock is powered up it displays 'Covert Entry' status. The service person myst obtain a 'Clear Covert Entry Code' to reset the ATM/Lock to normal operating mode. When the 'Covert Entry' status has been cleared the service person may use the current combination to open the ATM/Lock.

The following changes are required to CNCW95 software to support the Covert Entry Feature:

- o Add an entry to the 'Authorization Levels' file for 'Get Code to Clear Covert Entry'
- o Modify CAuthSvc to process the 'Get Code to Clear Covert Entry' authorization.
- o Create a class to process requests for 'Get Code to Clear Covert Entry'.
- o Add an item to the Supervisor Services Menu to allow an operator to request a Clear Covert Entry code.
- o Add an item to the 'Define Function Authorization Levels for Supervisor Services' panel for assigning Authorization Levels to the 'Get Code to Clear Covert Entry' function.
- o Modify CMainframe to invoke the 'Get Code to Clear Covert Entry' class when the Supervisor Services Menu requests it
- o Modify list_audit in cenconbe to translate the 'Get Gode to Clear Covert Entry' event.
- o Modify transdesc and lockdesc tables in cenconbe to allow list_auditkey to process the "Covert Entry" and "CE Cleared" events.

Flowchart blocks that depict operations affected by the "Covert Entry Feature" are shown in white. Other blocks are shown in gray.

Flowchart Index

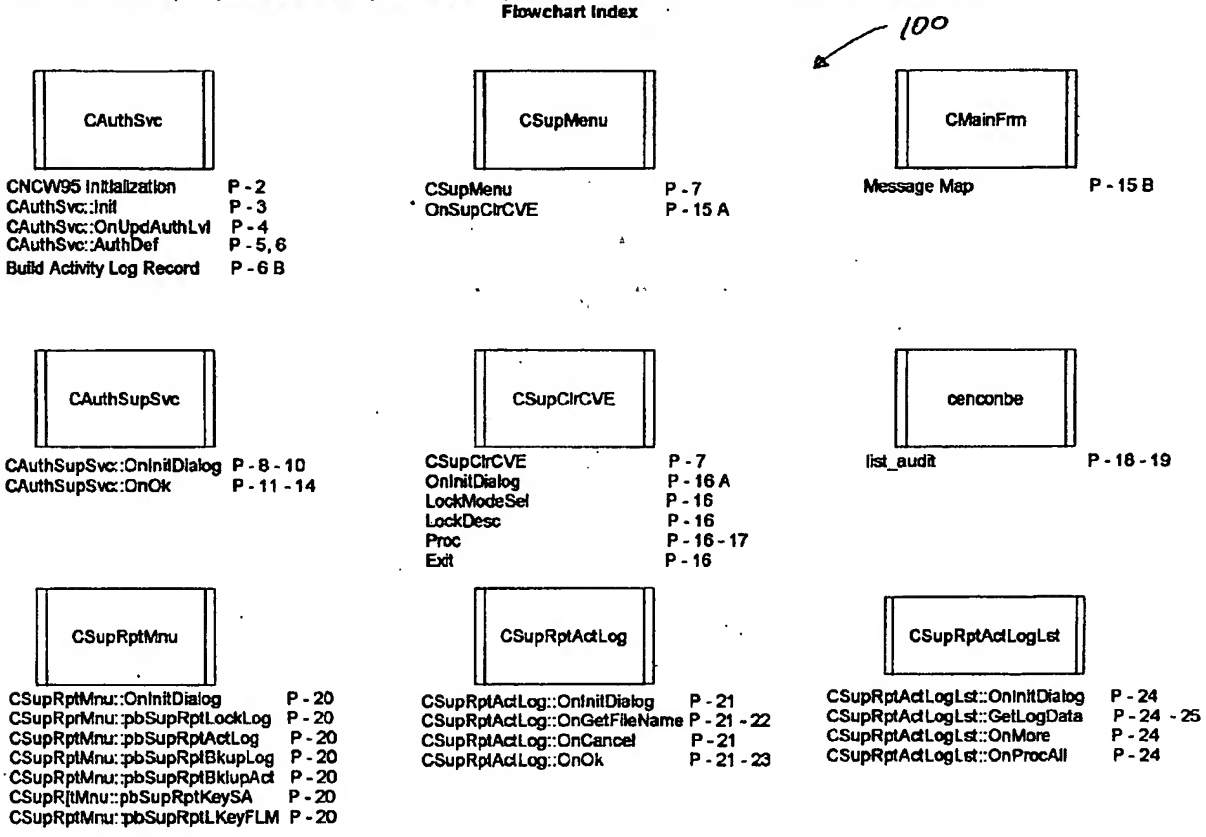


FIG. 16

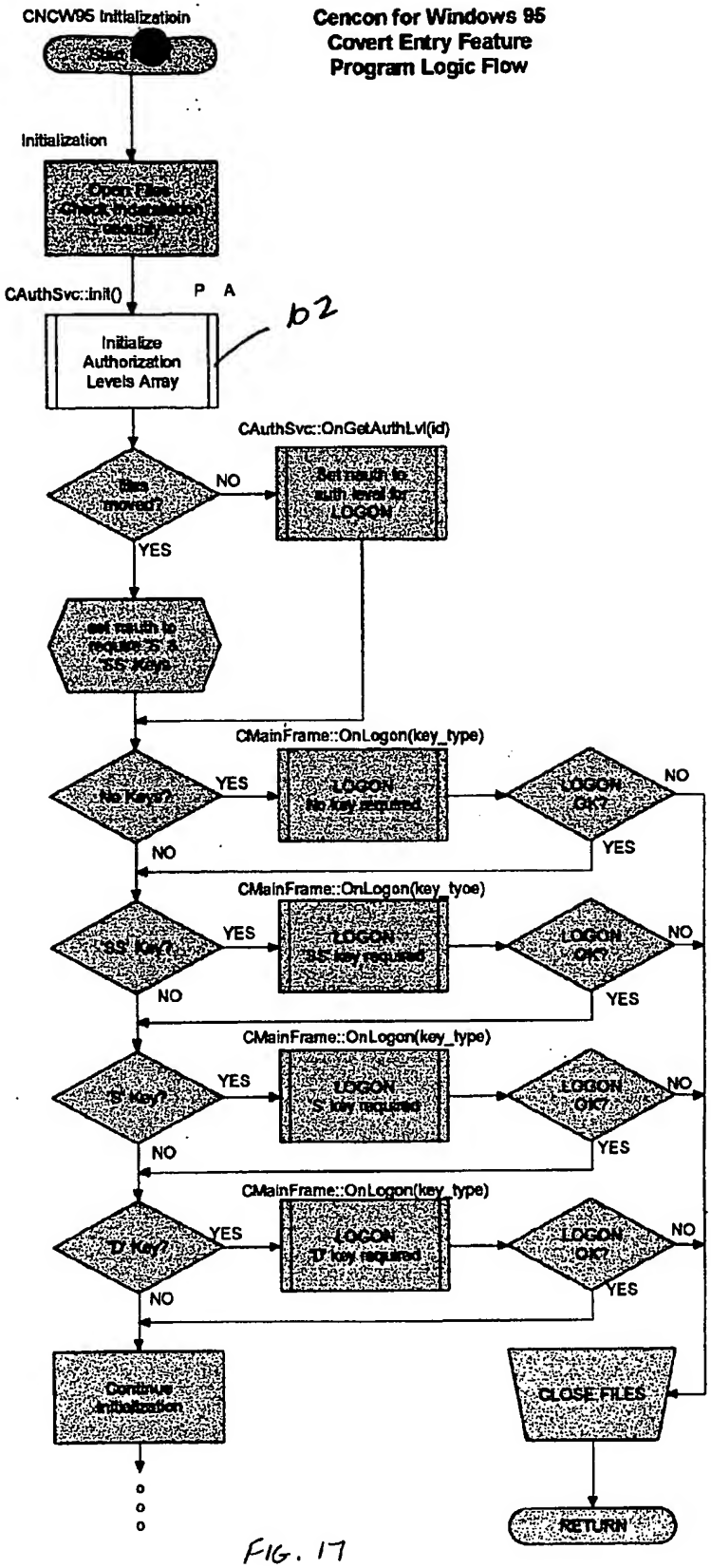
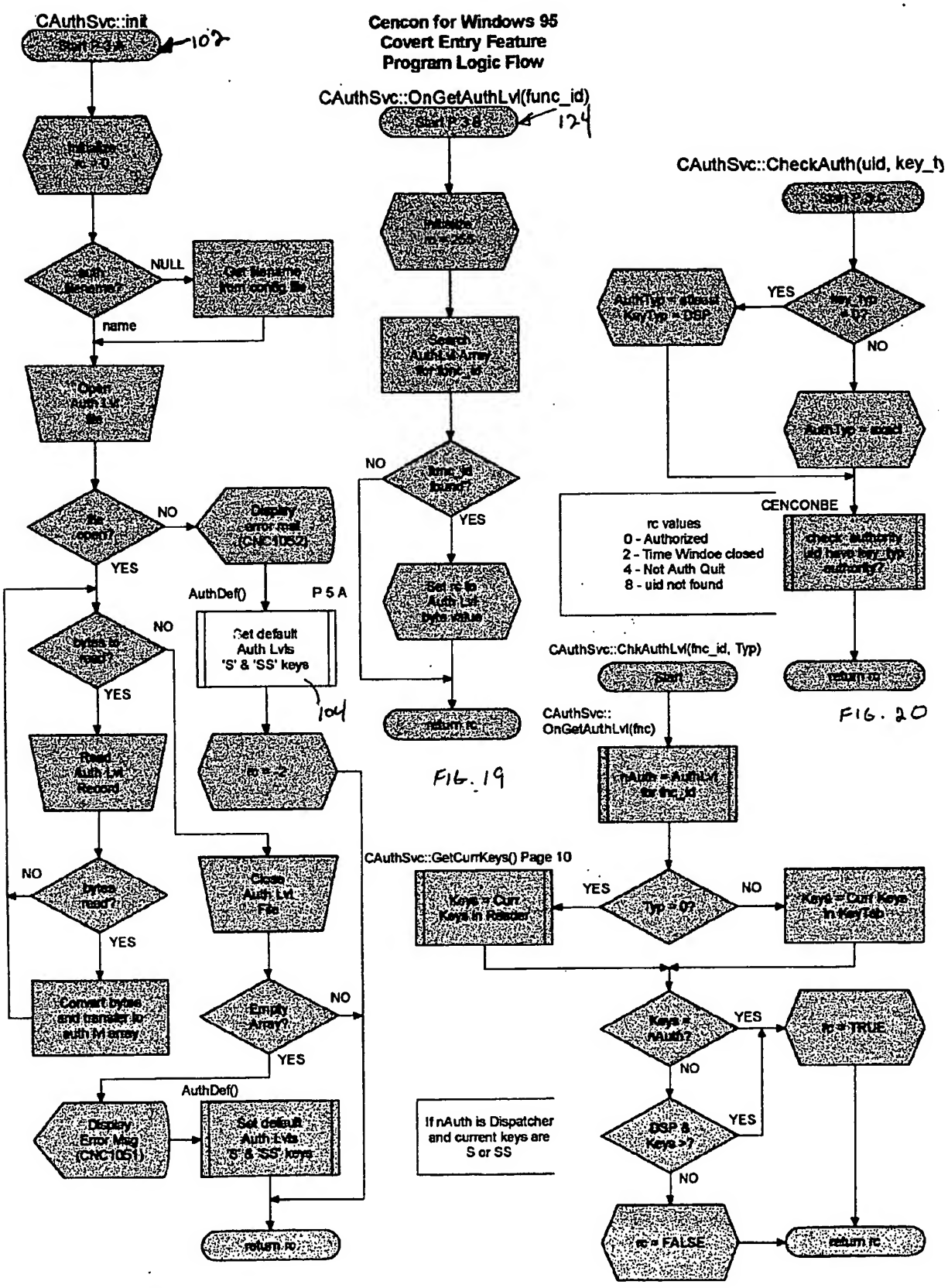

Cencon for Windows 95
Covert Entry Feature
Program Logic Flow
CNCW95 Initializatioin
Start
Initialization
Open Files
Check Installation
Security
CAuthSvc::Init()
P A
Initialize
Authorization
Levels Array
62
CAuthSvc::OnGetAuthLvl(id)
files moved?
NO
YES
Set result to auth level for LOGON
set result to require '5' & '35' keys
CMainFrame::OnLogon(key_type)
'No Keys'?
YES
LOGON
No key required
LOGON OK?
NO
YES
NO
CMainFrame::OnLogon(key_tyoe)
'35' key?
YES
LOGON
'35' key required
LOGON OK?
NO
YES
NO
CMainFrame::OnLogon(key_type)
'5' Key?
YES
LOGON
'5' key required
LOGON OK?
NO
YES
NO
CMainFrame::OnLogon(key_type)
'1' Key?
YES
LOGON
'1' key required
LOGON OK?
NO
YES
NO
Continue Initialization
CLOSE FILES
RETURN

FIG. 17


CAuthSvc::init
Start P 5 A
102
Initialize rc = 0
auth filename?
NULL
Get filename from config file
name
Open Auth Lvl file
file open?
NO
Display error msg (CNC1052)
YES
AuthDef()
P 5 A
bytes to read?
NO
YES
Set default Auth Lvls 'S' & 'SS' keys
104
Read Auth Lvl Record
rc = -2
bytes read?
NO
YES
Close Auth Lvl File
Convert bytes and transfer to auth M array
Empty Array?
NO
YES
AuthDef()
Display Error Msg (CNC1051)
Set default Auth Lvls 'S' & 'SS' keys
return rc
FIG. 18
Cencon for Windows 95
Covert Entry Feature
Program Logic Flow
CAuthSvc::OnGetAuthLvl(func_id)
Start P 3 B
124
Initialize rc = 255
Search Auth Lvl Array for func_id
func_id found?
NO
YES
Set rc to Auth Lvl byte value
return rc
FIG. 19
CAuthSvc::CheckAuth(uid, key_t)
Start P 5 C
key_typ = 0?
YES
AuthTyp = stored KeyTyp = DSP
NO
AuthTyp = stored
CENCONBE
rc values
0 - Authorized
2 - Time Window closed
4 - Not Auth Quit
8 - uid not found
Check authority uid have key_typ authority?
return rc
FIG. 20
CAuthSvc::ChkAuthLvl(fnc_id, Typ)
Start
CAuthSvc:: OnGetAuthLvl(fnc)
nAuth = AuthLvl for fnc_id
CAuthSvc::GetCurrKeys() Page 10
Typ = 0?
YES
NO
Keys = Curr Keys in Reader
Keys = Curr keys in KeyTab
Keys = nAuth?
YES
rc = TRUE
NO
DSP & Keys >?
YES
NO
rc = FALSE
return rc
If nAuth is Dispatcher and current keys are S or SS
FIG. 21

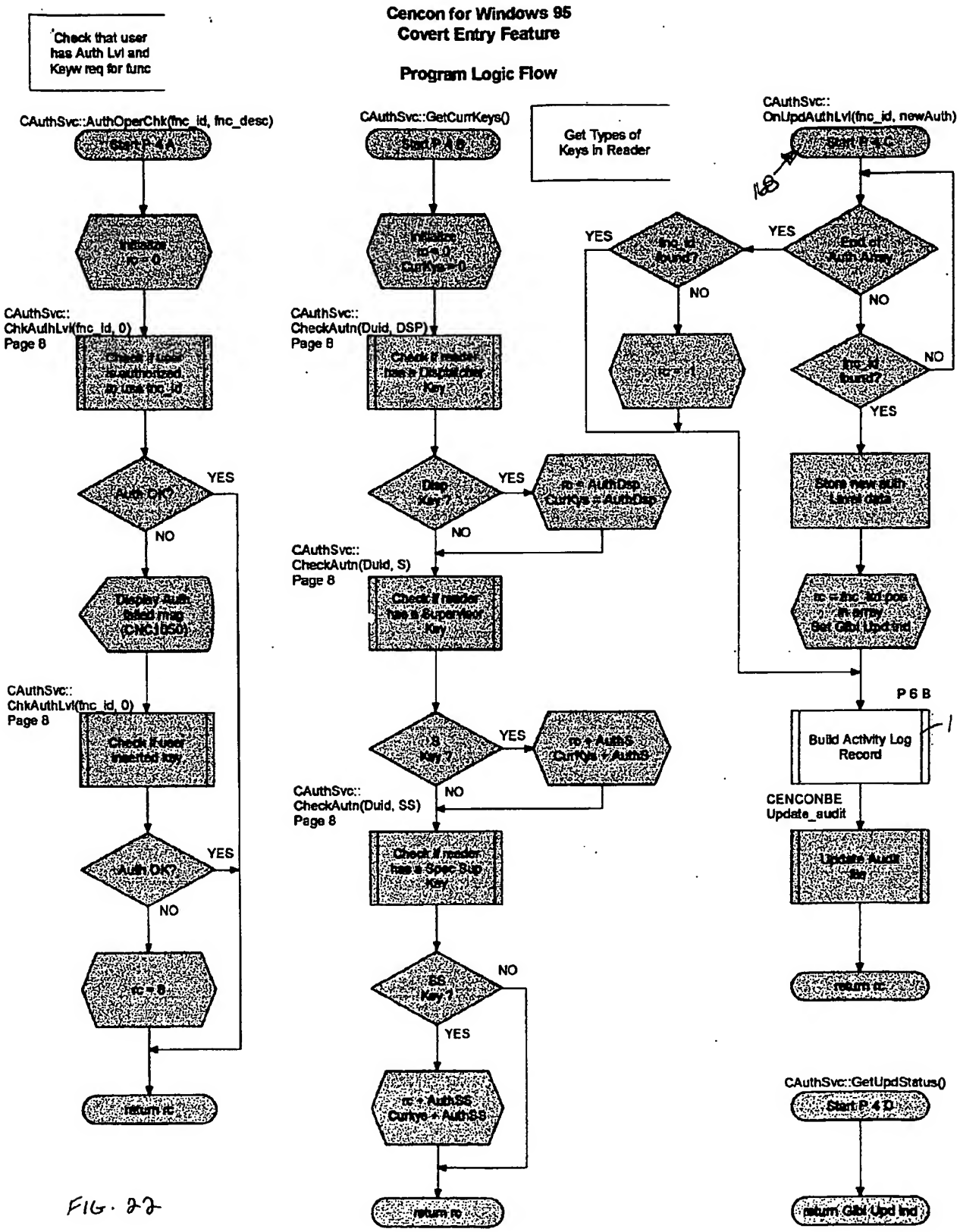


FIG. 22

FIG. 23

FIG. 24A

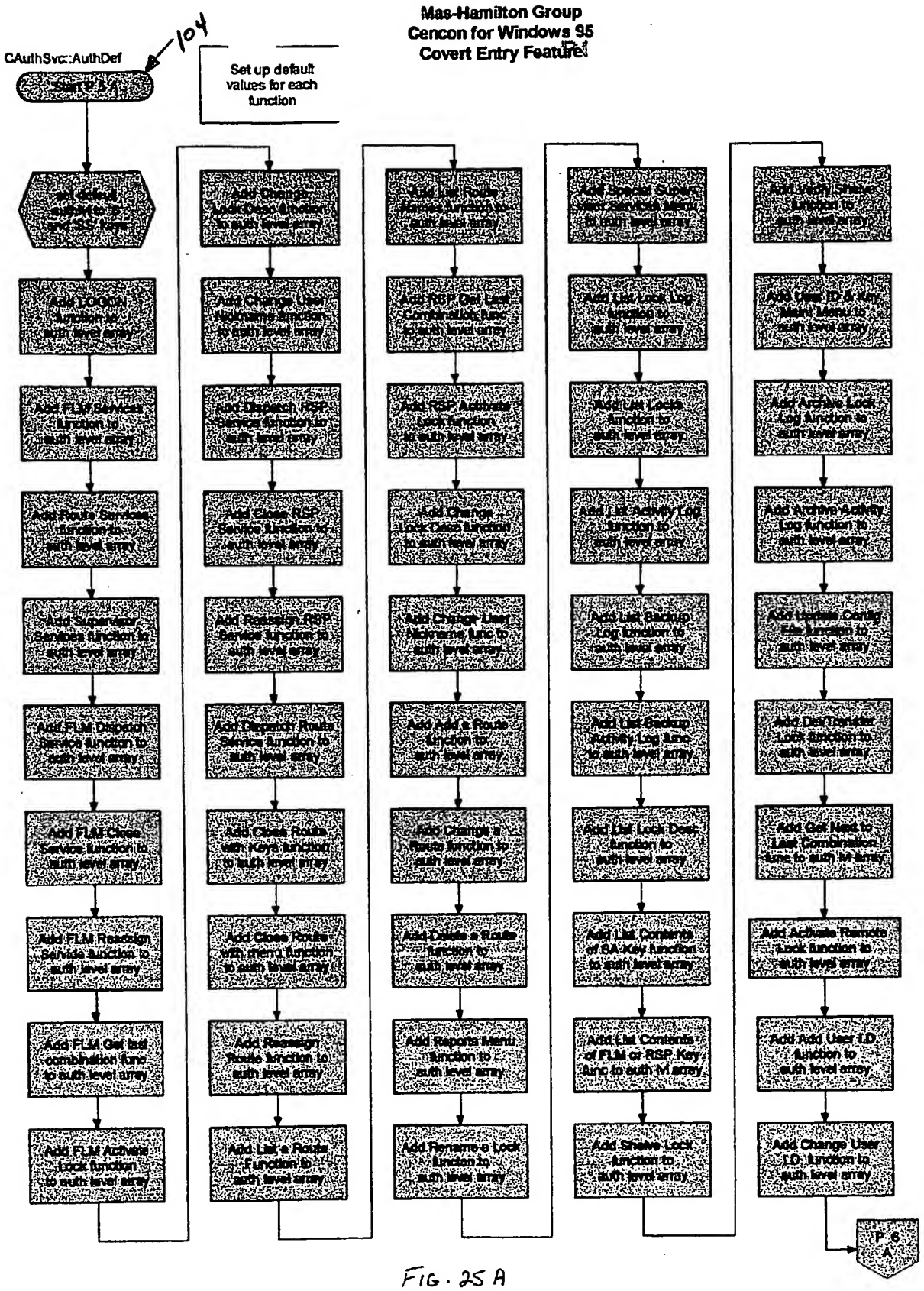

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
CAuthSvc::AuthDef
104
Start P 5 A
Set up default values for each function
Add LOGON function to auth level array
Add FLM Services function to auth level array
Add Route Services function to auth level array
Add Supervisor Services function to auth level array
Add FLM Disposition Service function to auth level array
Add FLM Close Service function to auth level array
Add FLM Reassign Service function to auth level array
Add FLM Get last combination func to auth level array
Add FLM Activate Lock function to auth level array
Add Change User Password function to auth level array
Add Change User Username function to auth level array
Add Dispense RSP Service function to auth level array
Add Close RSP Service function to auth level array
Add Reassign RSP Service function to auth level array
Add Dispense Route Service function to auth level array
Add Close Route with Keys function to auth level array
Add Close Route with menu function to auth level array
Add Reassign Route function to auth level array
Add List a Route Function to auth level array
Add List Route Names function to auth level array
Add RSP Get Last Combination func to auth level array
Add RSP Activate Lock function to auth level array
Add Change Lock Close function to auth level array
Add Change User Username func to auth level array
Add Add a Route function to auth level array
Add Change a Route function to auth level array
Add Delete a Route function to auth level array
Add Reports Menu function to auth level array
Add Rename a Lock function to auth level array
Add Supervisor Maint Services Menu to auth level array
Add List Lock Log function to auth level array
Add List Locks function to auth level array
Add List Activity Log function to auth level array
Add List Backup Log function to auth level array
Add List Backup Activity Log func to auth level array
Add List Lock Desk function to auth level array
Add List Contents of EA Key function to auth level array
Add List Contents of FLM or RSP Key func to auth M array
Add Shelve Lock function to auth level array
Add Modify Shelve Lock to auth level array
Add User ID & Key Maint Menu to auth level array
Add Archive Lock Log function to auth level array
Add Archive Activity Log function to auth level array
Add Update Config File function to auth level array
Add DeActivate Lock function to auth level array
Add Get Next to Last Combination func to auth M array
Add Activate Remote Lock function to auth level array
Add Add User I.D. function to auth level array
Add Change User I.D. function to auth level array
P 6 A

FIG. 25A

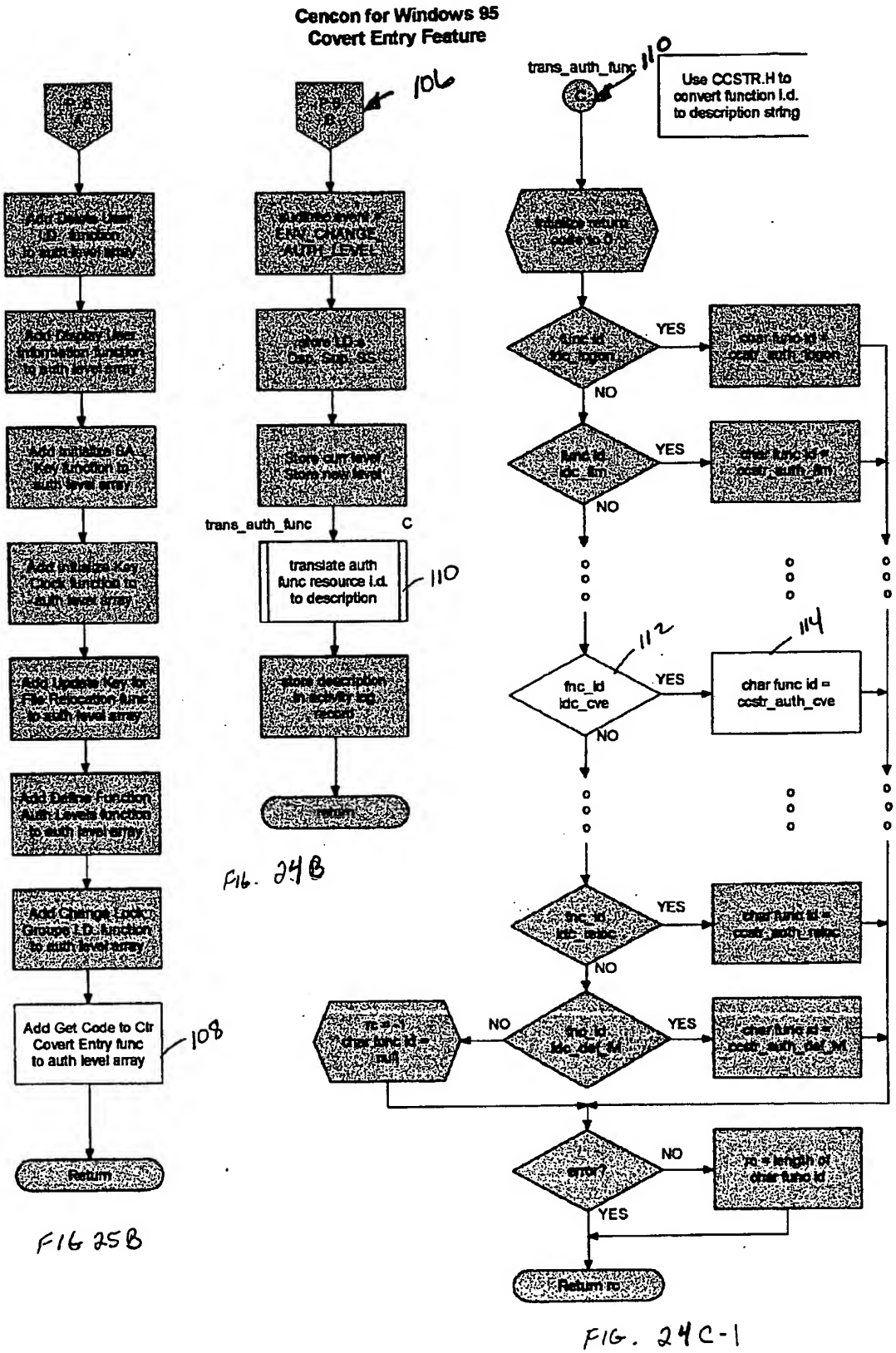


FIG 25B

FIG. 24B

FIG. 24C-1

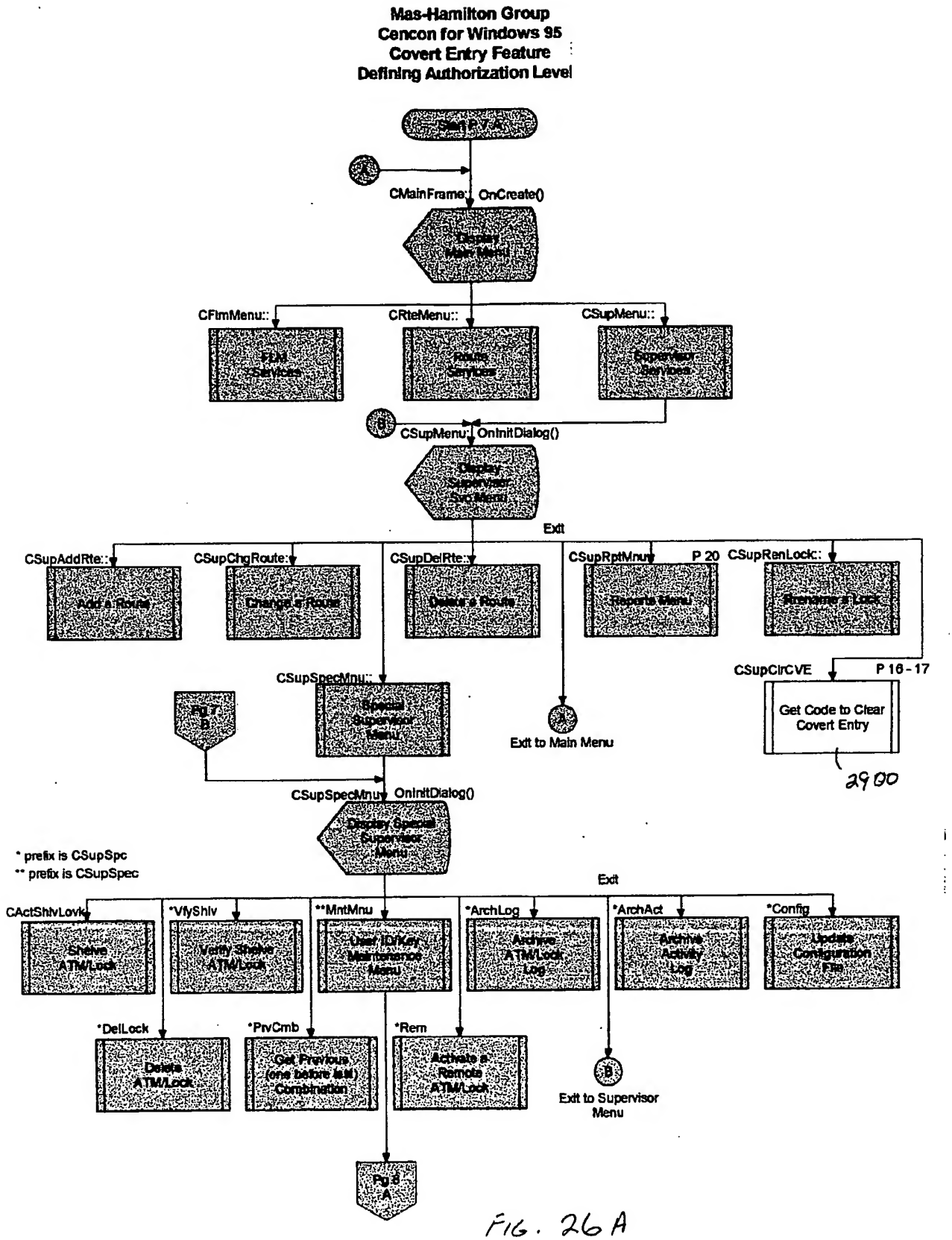

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
Defining Authorization Level
Start P 7 A
A
CMainFrame::OnCreate()
Display Main Menu
CFtmMenu::
CRteMenu::
CSupMenu::
FDM Services
Route Services
Supervisor Services
B
CSupMenu::OnInitDialog()
Display Supervisor Svc Menu
Exit
CSupAddRte::
CSupChgRoute::
CSupDelRte::
CSupRptMnu
P 20
CSupRenLock::
Add a Route
Change a Route
Delete a Route
Reports Menu
Rename a Lock
CSupClrCVE
P 16 - 17
Get Code to Clear Covert Entry
2900
CSupSpecMnu::
Special Supervisor Menu
Pg 7 B
A
Exit to Main Menu
CSupSpecMnu::OnInitDialog()
Display Special Supervisor Menu
* prefix is CSupSpc
** prefix is CSupSpec
Exit
CActShlvLovk
*VfyShlv
**MntMnu
*ArchLog
*ArchAct
*Config
Shelve ATM/Lock
Verify Shelve ATM/Lock
User ID/Key Maintenance Menu
Archive ATM/Lock Log
Archive Activity Log
Update Configuration File
*DelLock
*PrvCmb
*Rem
Delete ATM/Lock
Get Previous (one before last) Combination
Activate a Remote ATM/Lock
B
Exit to Supervisor Menu
Pg 8 A

FIG. 26A

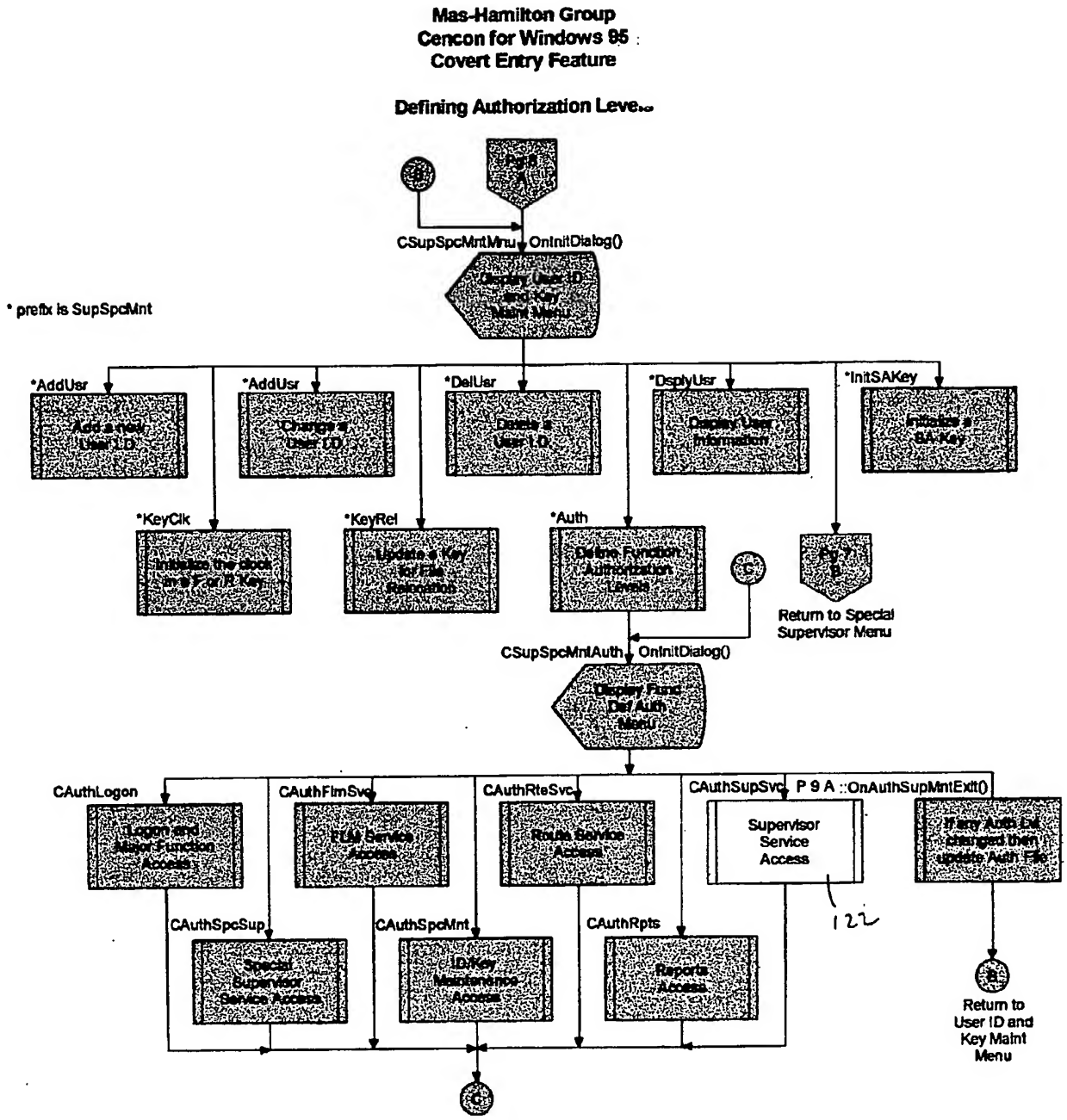

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
Defining Authorization Leve...
* prefix is SupSpcMnt
CSupSpcMntMnu OnInitDialog()
*AddUsr
*AddUsr
*DelUsr
*DsplyUsr
*InitSAKey
*KeyClk
*KeyRel
*Auth
Return to Special Supervisor Menu
CSupSpcMntAuth OnInitDialog()
CAuthLogon
CAuthFlmSvc
CAuthRteSvc
CAuthSupSvc
P 9 A ::OnAuthSupMntExit()
Supervisor Service Access
CAuthSpcSup
CAuthSpcMnt
CAuthRpts
Return to User ID and Key Maint Menu

FIG. 26 B

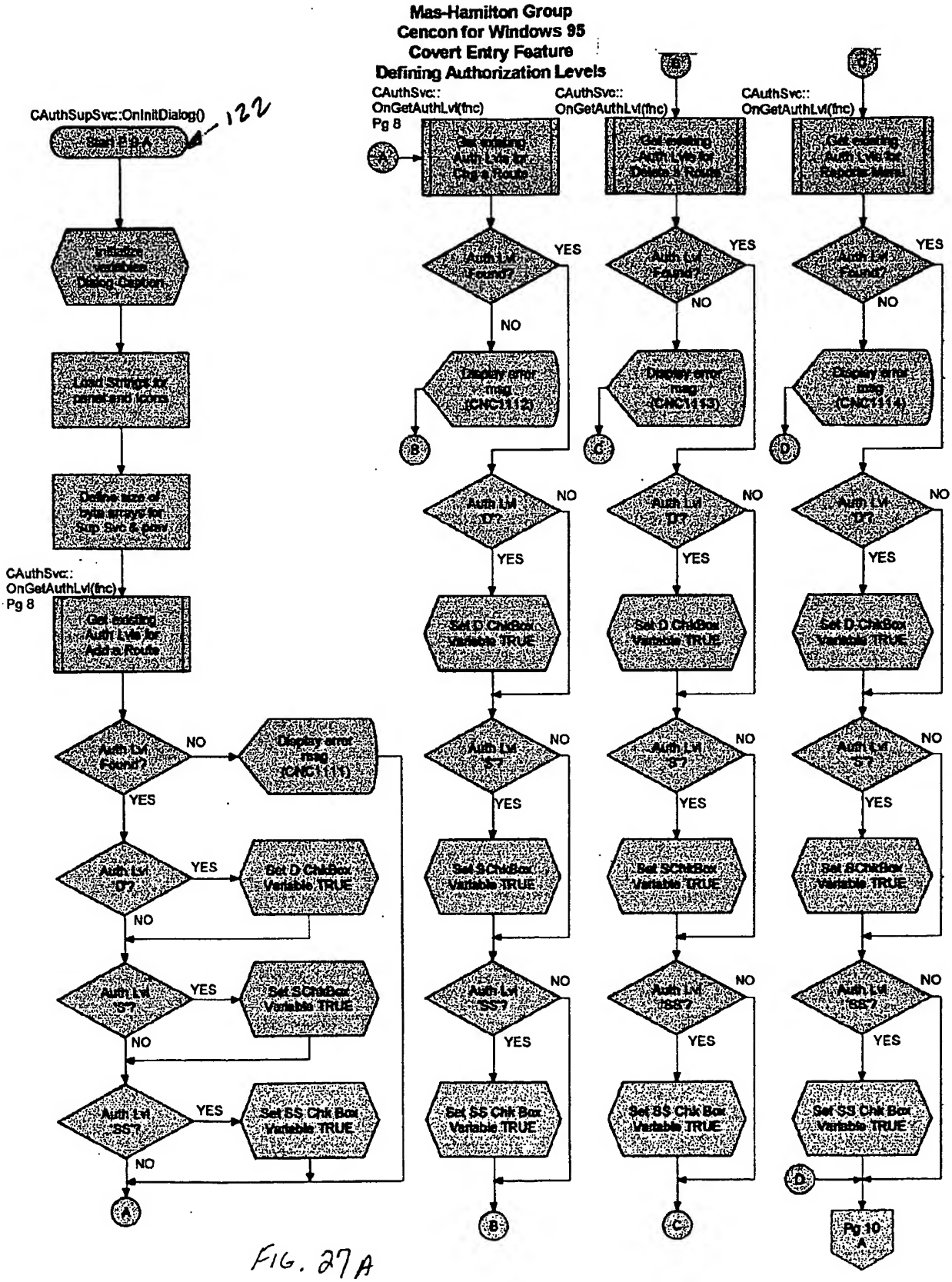

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
Defining Authorization Levels
CAuthSupSvc::OnInitDialog()
122
Start 2 of 4
Initialize variables, Dialog Caption
Load Strings for names and icons
Define size of byte arrays for Sup Svc & pass
CAuthSvc:: OnGetAuthLvl(fnc) Pg 8
Get existing Auth Lvls for Add a Route
Auth Lvl Found?
NO
Display error msg (CNC1111)
YES
Auth Lvl 'D'?
YES
Set D ChkBox Variable TRUE
NO
Auth Lvl 'S'?
YES
Set S ChkBox Variable TRUE
NO
Auth Lvl 'SS'?
YES
Set SS Chk Box Variable TRUE
NO
A
CAuthSvc:: OnGetAuthLvl(fnc) Pg 8
Get existing Auth Lvls for Add a Route
Auth Lvl Found?
YES
NO
Display error msg (CNC1112)
B
Auth Lvl 'D'?
NO
YES
Set D ChkBox Variable TRUE
Auth Lvl 'S'?
NO
YES
Set S ChkBox Variable TRUE
Auth Lvl 'SS'?
NO
YES
Set SS Chk Box Variable TRUE
B
CAuthSvc:: OnGetAuthLvl(fnc)
Get existing Auth Lvls for Delete a Route
Auth Lvl Found?
YES
NO
Display error msg (CNC1113)
C
Auth Lvl 'D'?
NO
YES
Set D ChkBox Variable TRUE
Auth Lvl 'S'?
NO
YES
Set S ChkBox Variable TRUE
Auth Lvl 'SS'?
NO
YES
Set SS Chk Box Variable TRUE
C
CAuthSvc:: OnGetAuthLvl(fnc)
Get existing Auth Lvls for Change Route
Auth Lvl Found?
YES
NO
Display error msg (CNC1114)
D
Auth Lvl 'D'?
NO
YES
Set D ChkBox Variable TRUE
Auth Lvl 'S'?
NO
YES
Set S ChkBox Variable TRUE
Auth Lvl 'SS'?
NO
YES
Set SS Chk Box Variable TRUE
D
Pg 10 A

FIG. 27A

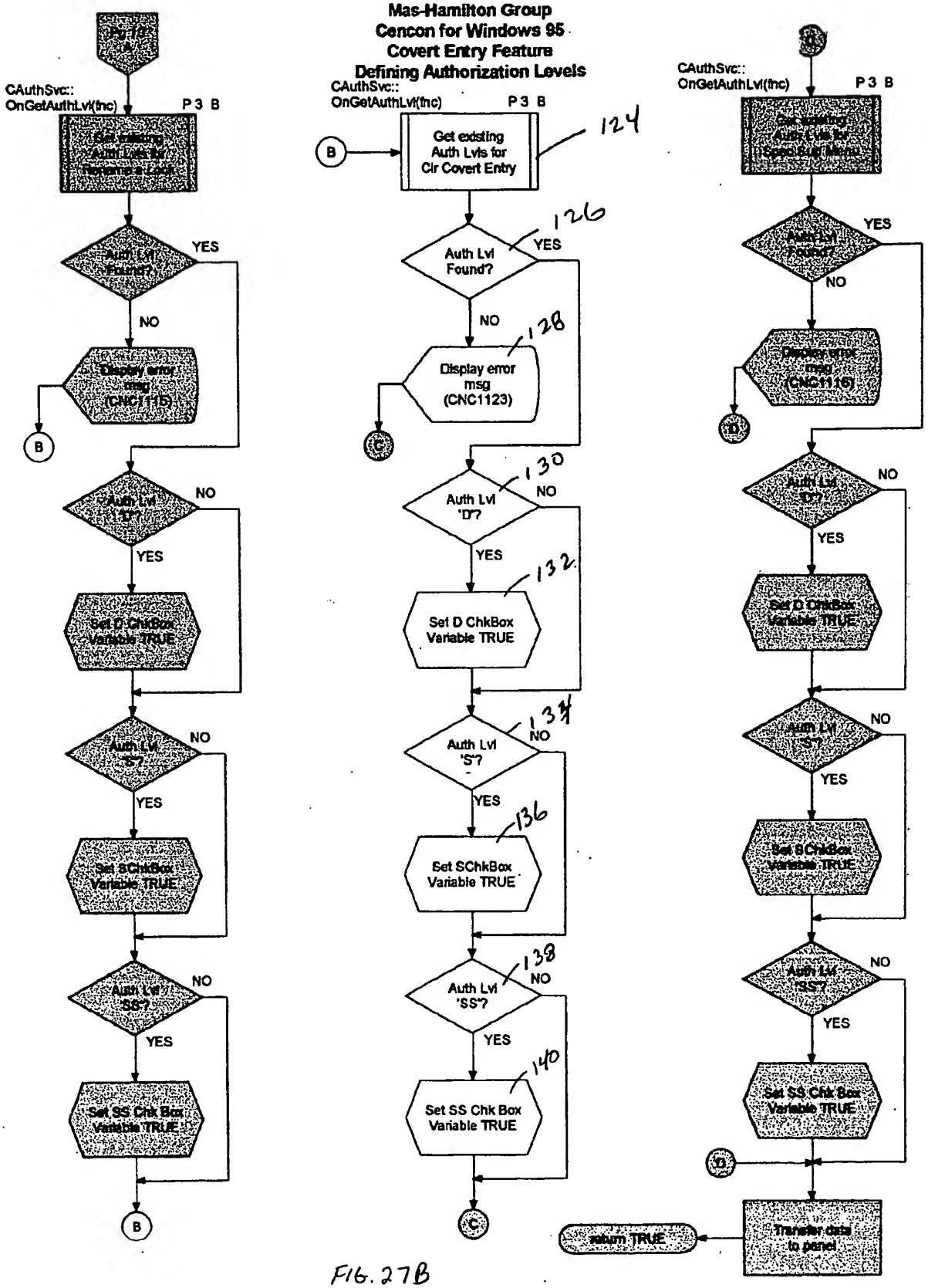

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
Defining Authorization Levels
CAuthSvc::
OnGetAuthLvl(hnc)
P 3 B
Get existing Auth Lvls for Clr Covert Entry
124
Auth Lvl Found?
YES
NO
126
Display error msg (CNC1123)
128
Auth Lvl 'D'?
130
Set D ChkBox Variable TRUE
132
Auth Lvl 'S'?
134
Set SChkBox Variable TRUE
136
Auth Lvl 'SS'?
138
Set SS Chk Box Variable TRUE
140
B
C
D
return TRUE
Transfer data to panel

FIG. 27B

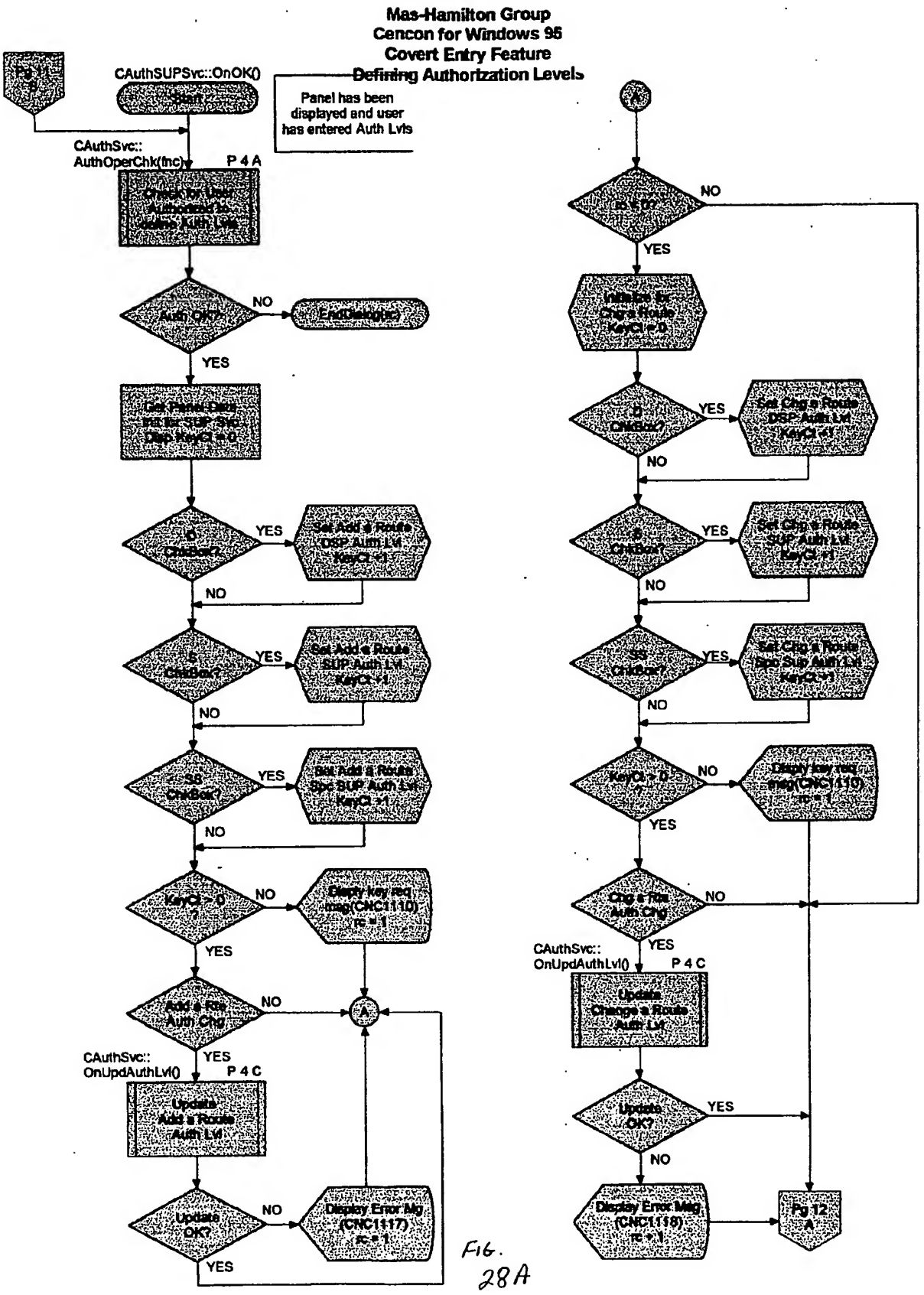

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
Defining Authorization Levels
Panel has been displayed and user has entered Auth Lvls
CAuthSUPSvc::OnOK()
Start
CAuthSvc::
AuthOperChk(thc)
P 4 A
Auth OK?
NO
YES
Set Panel Data and for SUP Svc: Disp KeyCt = 0
Add a Route ChkBox?
Set Add a Route DSP Auth Lvl KeyCt +1
Set Add a Route SUP Auth Lvl KeyCt +1
Set Add a Route Spc SUP Auth Lvl KeyCt +1
KeyCt > 0
Display key req msg(CNC1110) rc = 1
Add a Rte Auth Chg
CAuthSvc::
OnUpdAuthLvl()
P 4 C
Update Add a Route Auth Lvl
Update OK?
Display Error Msg (CNC1117) rc = 1
A
Chg a Rte Auth Chg
Initialize for Chg a Route KeyCt = 0
Set Chg a Route DSP Auth Lvl KeyCt +1
Set Chg a Route SUP Auth Lvl KeyCt +1
Set Chg a Route Spc SUP Auth Lvl KeyCt +1
Update Change a Route Auth Lvl
Display Error Msg (CNC1118) rc = 1
Pg 12 A

Fig. 28A

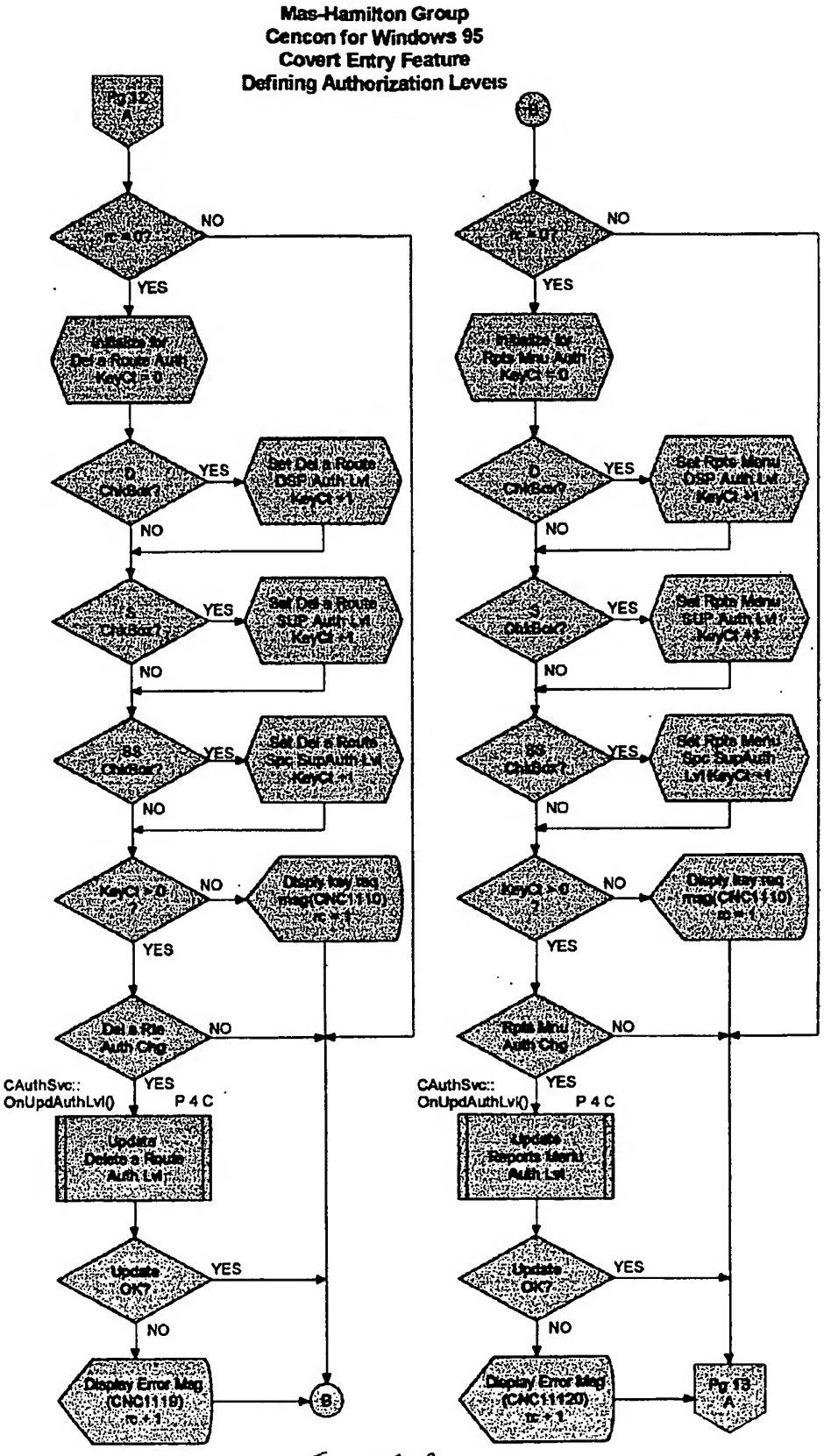

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
Defining Authorization Levels
Pg 12 A
rc = 0?
NO
YES
Initialize for Del a Route Auth KeyCt = 0
0 ChkBox?
YES
Set Del a Route DSP Auth Lvl KeyCt +1
NO
5 ChkBox?
YES
Set Del a Route SUP Auth Lvl KeyCt +1
NO
SS ChkBox?
YES
Set Del a Route Spc SupAuth Lvl KeyCt +1
NO
KeyCt > 0 ?
NO
Display key req msg(CNC1110) rc = 1
YES
Del a Rte Auth Chg
NO
YES
CAuthSvc:: OnUpdAuthLvl()
P 4 C
Update Delete a Route Auth Lvl
Update OK?
YES
NO
Display Error Msg (CNC1118) rc = 1
B
Pg 13 A
rc = 0?
NO
YES
Initialize for Rpts Menu Auth KeyCt = 0
0 ChkBox?
YES
Set Rpts Menu DSP Auth Lvl KeyCt +1
NO
5 ChkBox?
YES
Set Rpts Menu SUP Auth Lvl KeyCt +1
NO
SS ChkBox?
YES
Set Rpts Menu Spc SupAuth Lvl KeyCt = 1
NO
KeyCt > 0 ?
NO
Display key req msg(CNC1110) rc = 1
YES
Rpts Menu Auth Chg
NO
YES
CAuthSvc:: OnUpdAuthLvl()
P 4 C
Update Reports Menu Auth Lvl
Update OK?
YES
NO
Display Error Msg (CNC11120) rc = 1
Pg 13 A

FIG. 28 B

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
Defining Authorization Levels

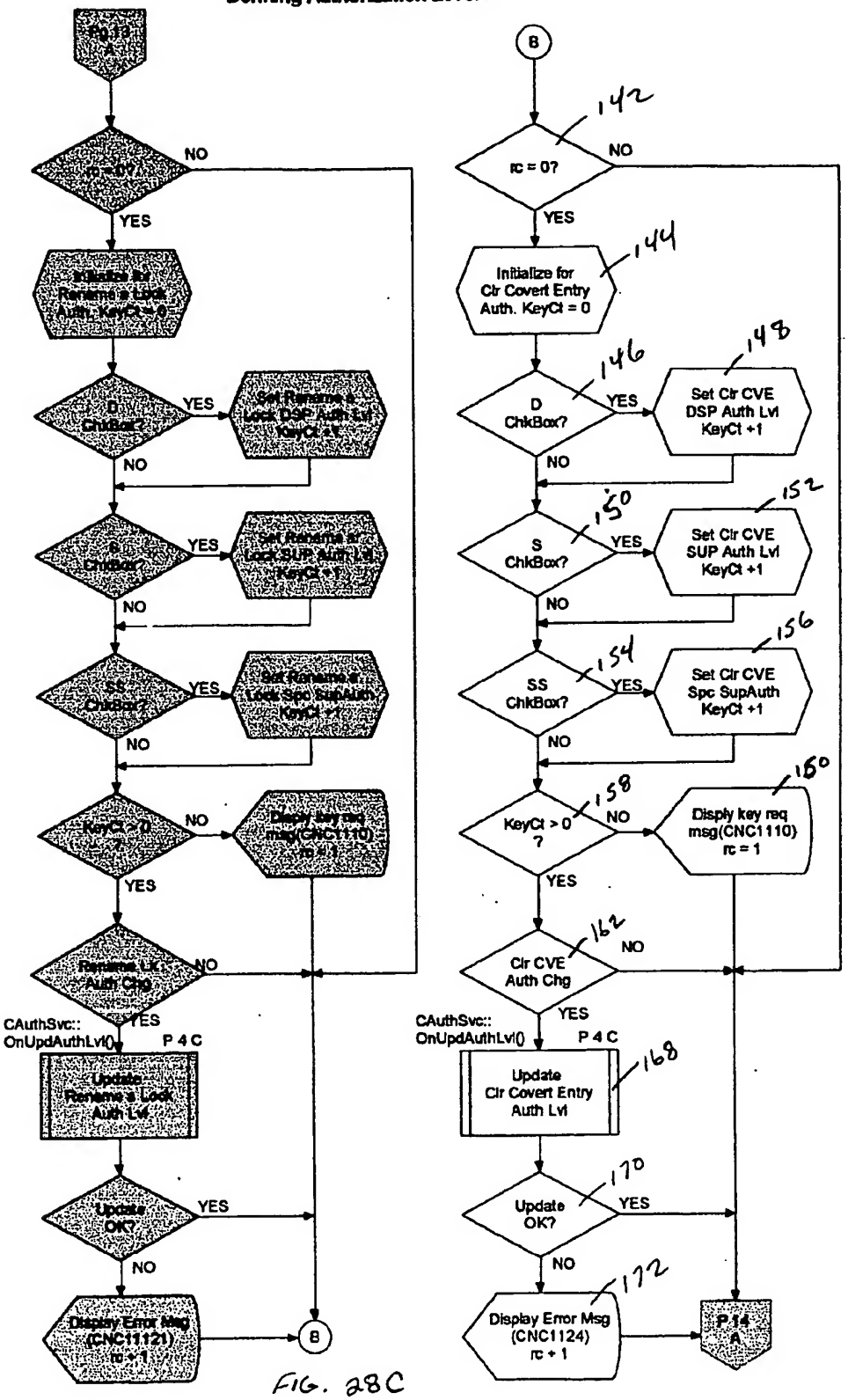


FIG. 28C

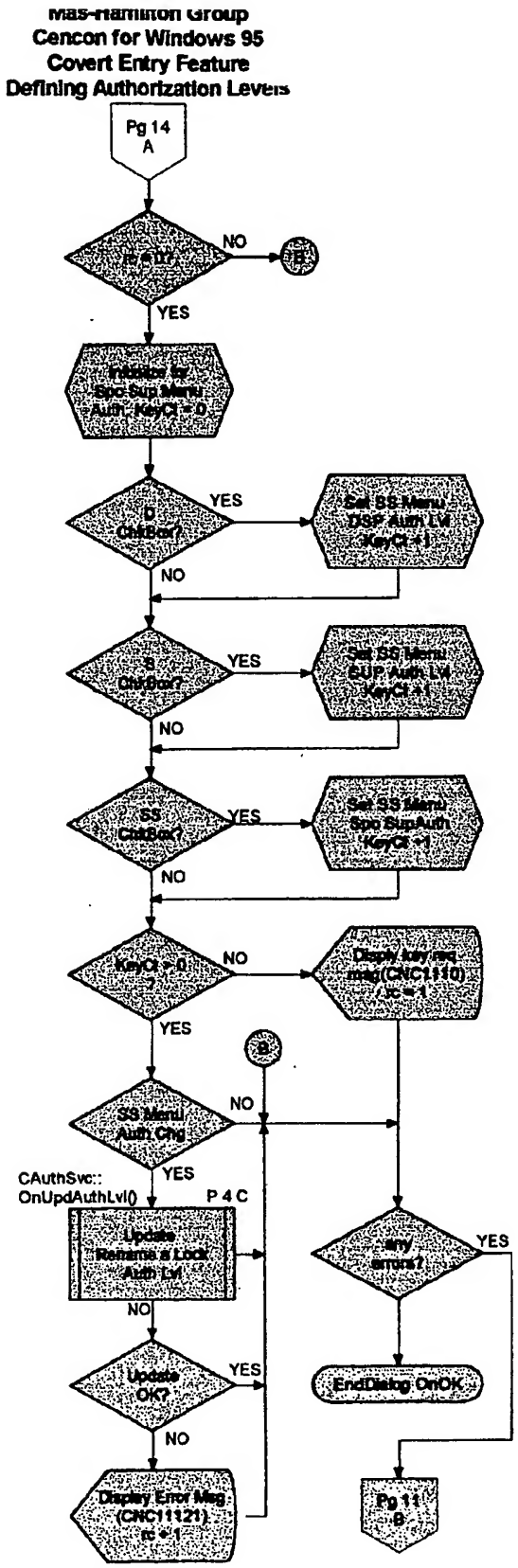

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
Defining Authorization Levels
Pg 14
A
rc = 0?
NO
YES
Initialize by
Spc Sup Menu
=Auth_KeyCt = 0
D
ChkBox?
YES
NO
S
ChkBox?
YES
NO
SS
ChkBox?
YES
NO
KeyCt > 0
?
NO
YES
SS Menu
Auth Chg
YES
NO
CAuthSvc::
OnUpdAuthLvl()
P 4 C
Update
Rename a Lock
Auth Lvl
NO
Update
OK?
YES
NO
Display Error Msg
(CNC11121)
rc = 1
any
errors?
YES
EndDialog OnOK
Pg 11
B

FIG 2B D

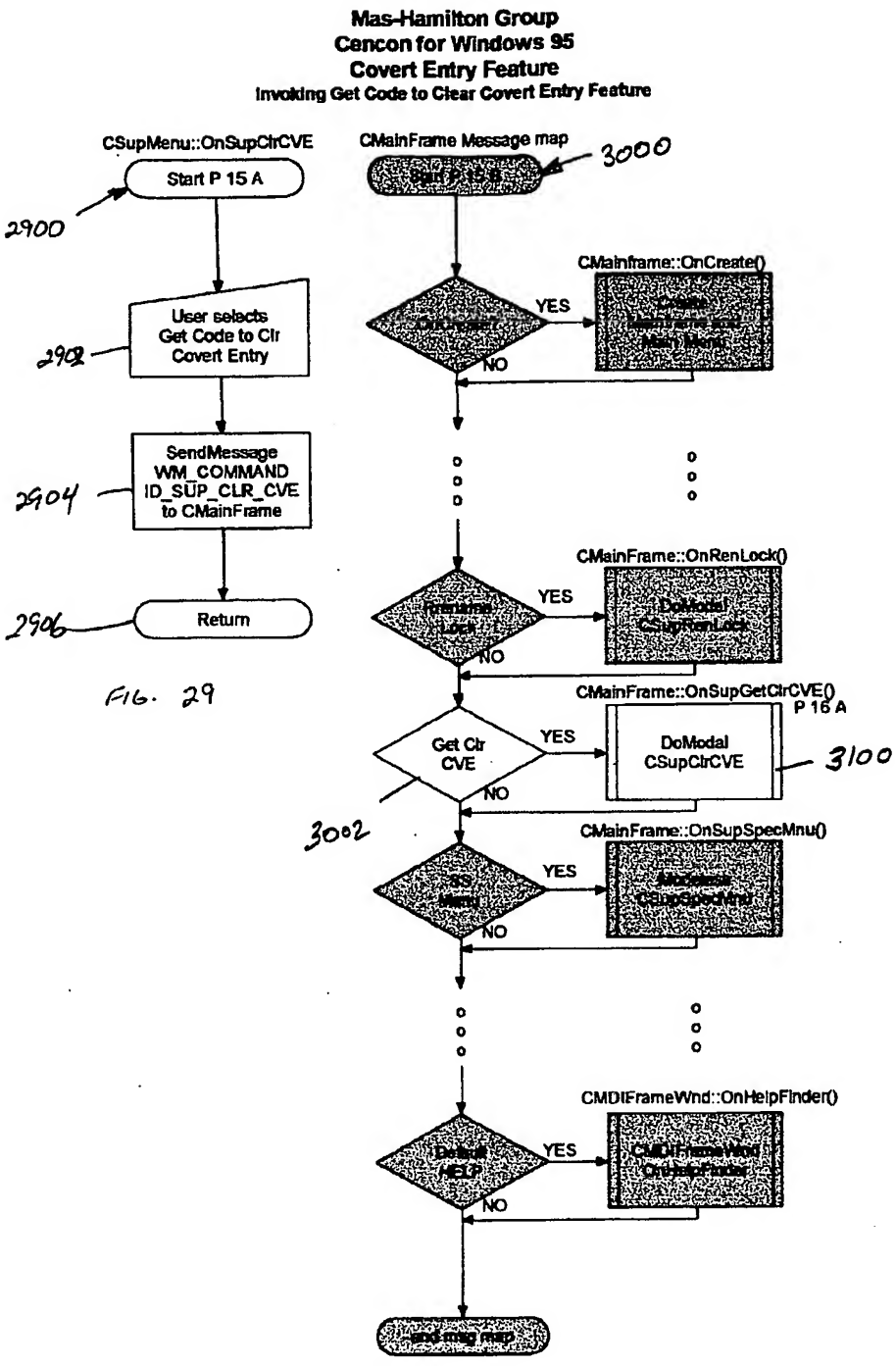

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
Invoking Get Code to Clear Covert Entry Feature
CSupMenu::OnSupClrCVE
Start P 15 A
2900
User selects Get Code to Clr Covert Entry
2902
SendMessage WM_COMMAND ID_SUP_CLR_CVE to CMainFrame
2904
Return
2906
CMainFrame Message map
3000
CMainframe::OnCreate()
YES
NO
CMainFrame::OnRenLock()
YES
NO
CMainFrame::OnSupGetClrCVE()
P 16 A
Get Clr CVE
YES
DoModal CSupClrCVE
3100
NO
3002
CMainFrame::OnSupSpecMnu()
YES
NO
CMDIFrameWnd::OnHelpFinder()
YES
NO

FIG. 29

FIG. 30

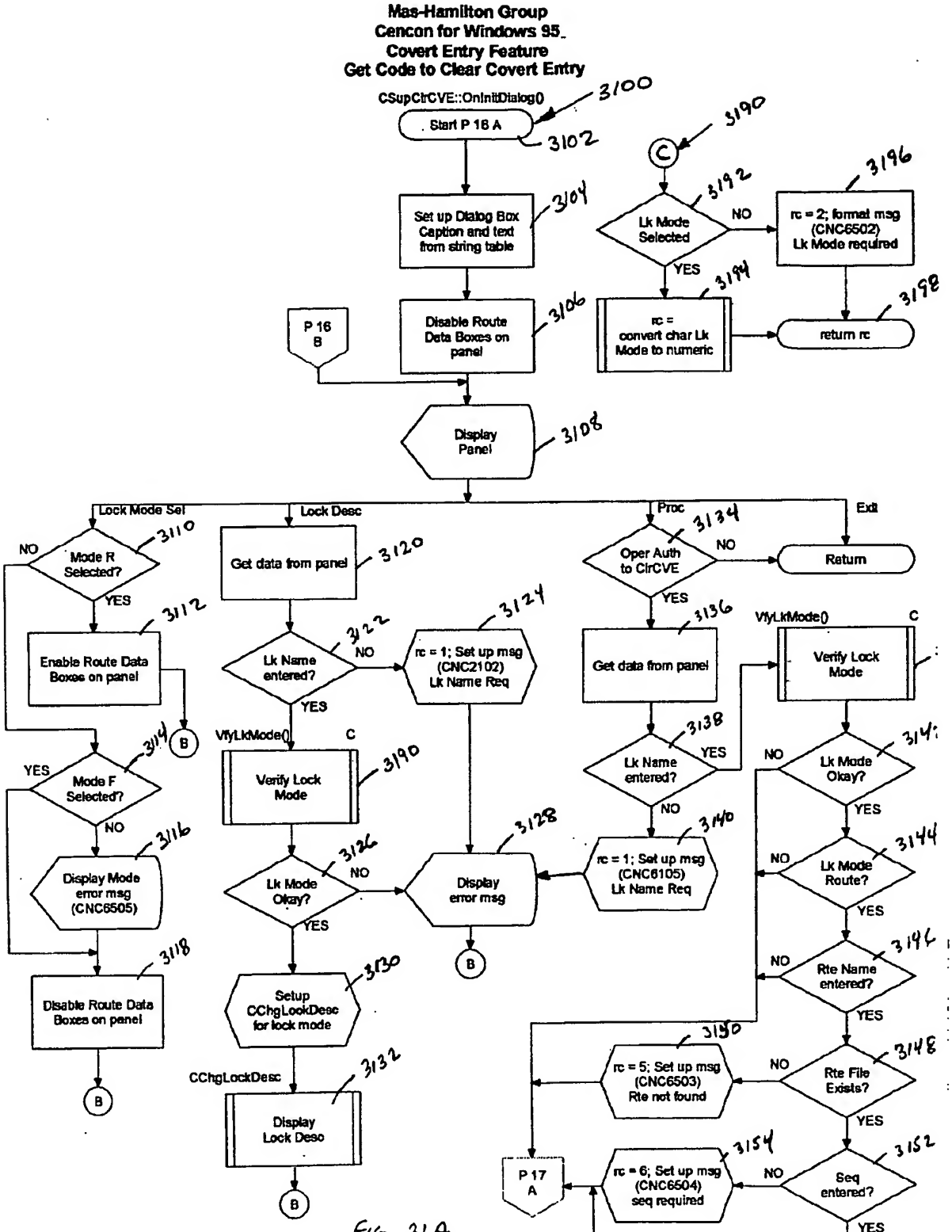

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
Get Code to Clear Covert Entry
CSupCtrCVE::OnInitDialog()
3100
Start P 16 A
3102
Set up Dialog Box Caption and text from string table
3104
Disable Route Data Boxes on panel
3106
P 16 B
Display Panel
3108
C
3190
Lk Mode Selected
3192
NO
YES
rc = 2; format msg (CNC6502) Lk Mode required
3196
rc = convert char Lk Mode to numeric
3194
return rc
3198
Lock Mode Sel
Lock Desc
Proc
Exit
Mode R Selected?
3110
NO
YES
Enable Route Data Boxes on panel
3112
Get data from panel
3120
Lk Name entered?
3122
rc = 1; Set up msg (CNC2102) Lk Name Req
3124
Oper Auth to CtrCVE
3134
Return
Get data from panel
3136
VfyLkMode()
C
Verify Lock Mode
Lk Name entered?
3138
Mode F Selected?
3114
Display Mode error msg (CNC6505)
3116
Disable Route Data Boxes on panel
3118
VfyLkMode()
C
Verify Lock Mode
3190
Lk Mode Okay?
3126
Setup CChgLookDesc for lock mode
3130
CChgLockDesc
Display Lock Desc
3132
Display error msg
3128
rc = 1; Set up msg (CNC6105) Lk Name Req
3140
Lk Mode Okay?
Lk Mode Route?
Rte Name entered?
Rte File Exists?
Seq entered?
rc = 5; Set up msg (CNC6503) Rte not found
3150
rc = 6; Set up msg (CNC6504) seq required
3154
P 17 A
B

FIG. 31A

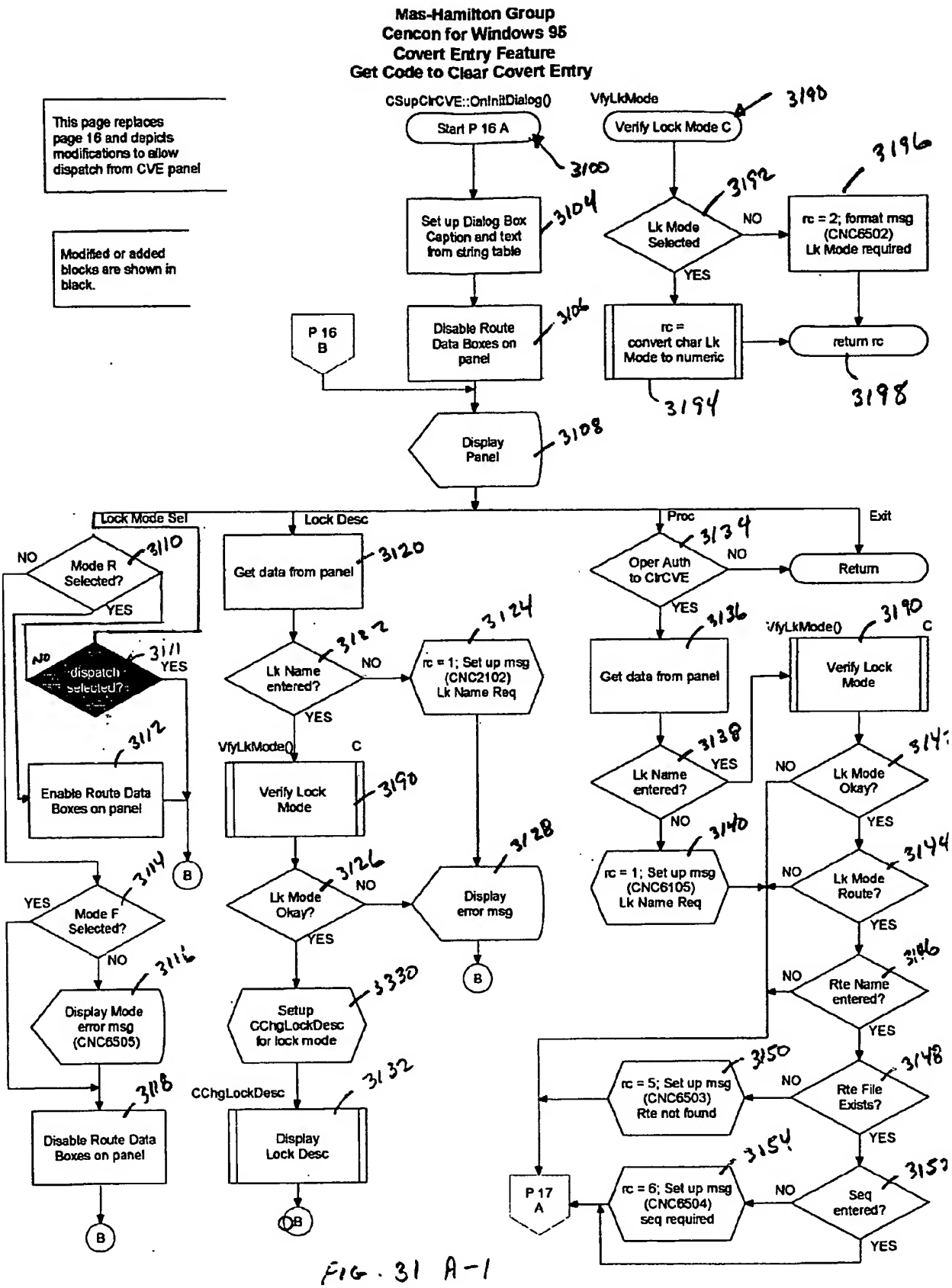

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
Get Code to Clear Covert Entry
This page replaces page 16 and depicts modifications to allow dispatch from CVE panel
Modified or added blocks are shown in black.
CSupClrCVE::OnInitDialog()
Start P 16 A
3100
Set up Dialog Box Caption and text from string table
3104
P 16 B
Disable Route Data Boxes on panel
3106
Display Panel
3108
VfyLkMode
Verify Lock Mode C
3190
Lk Mode Selected
3192
NO
YES
rc = 2; format msg (CNC6502) Lk Mode required
3196
rc = convert char Lk Mode to numeric
3194
return rc
3198
Lock Mode Sel
Lock Desc
Proc
Exit
Mode R Selected?
3110
NO
YES
dispatch selected?
3111
Enable Route Data Boxes on panel
3112
Mode F Selected?
3114
YES
NO
Display Mode error msg (CNC6505)
3116
Disable Route Data Boxes on panel
3118
B
Get data from panel
3120
Lk Name entered?
3122
NO
YES
rc = 1; Set up msg (CNC2102) Lk Name Req
3124
VfyLkMode()
C
Verify Lock Mode
3190
Lk Mode Okay?
3126
NO
YES
Display error msg
3128
B
Setup CChgLockDesc for lock mode
3330
CChgLockDesc
Display Lock Desc
3132
B
Oper Auth to ClrCVE
3134
NO
YES
Return
Get data from panel
3136
Lk Name entered?
3138
YES
NO
rc = 1; Set up msg (CNC6105) Lk Name Req
3140
VfyLkMode()
C
Verify Lock Mode
3190
Lk Mode Okay?
3142
NO
YES
Lk Mode Route?
3144
NO
YES
Rte Name entered?
3146
NO
YES
Rte File Exists?
3148
NO
YES
rc = 5; Set up msg (CNC6503) Rte not found
3150
P 17 A
rc = 6; Set up msg (CNC6504) seq required
3154
Seq entered?
3152
NO
YES

FIG. 31 A-1

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
Get Code to Clear Covert Entry

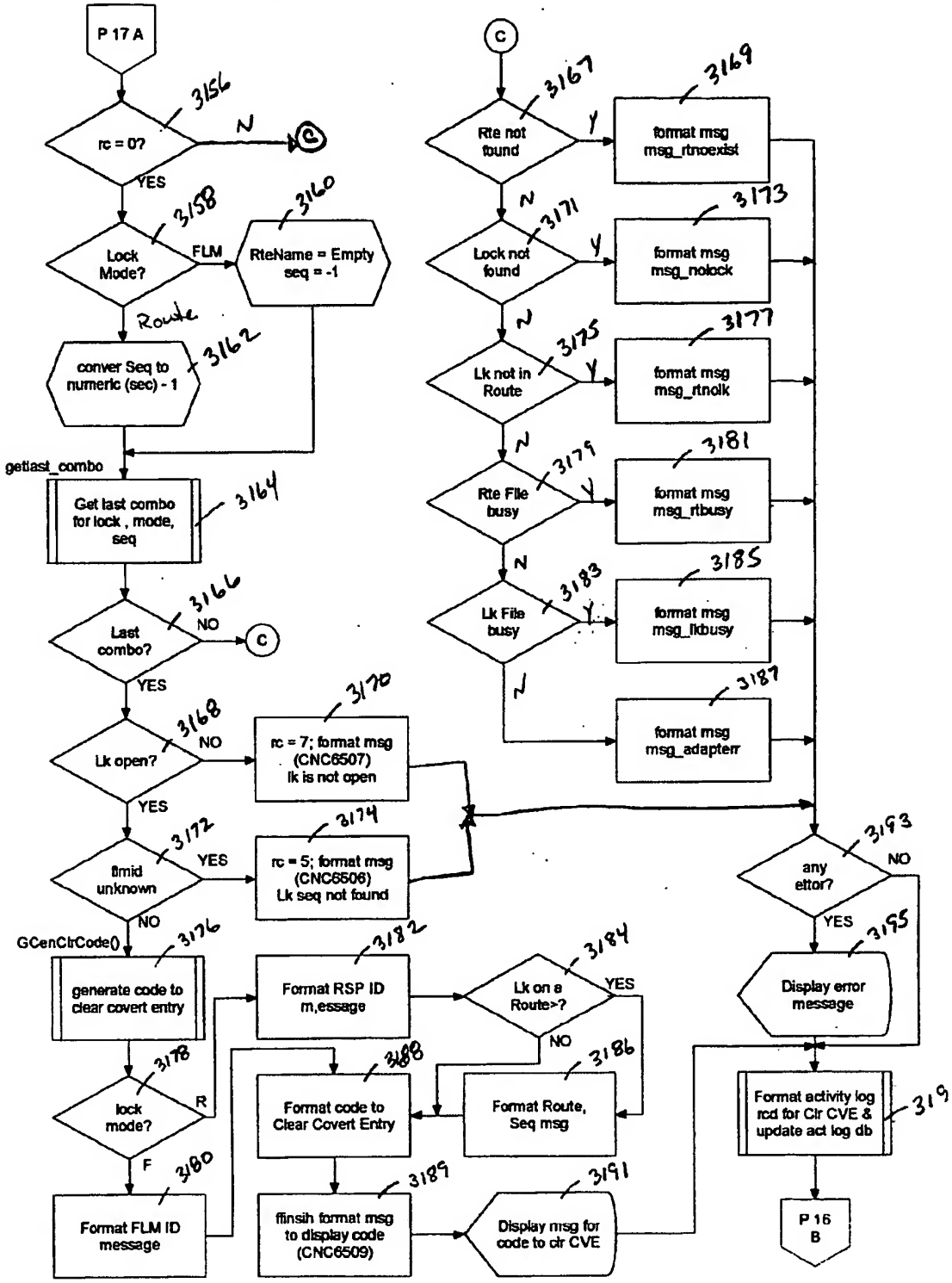


FIG. 31B

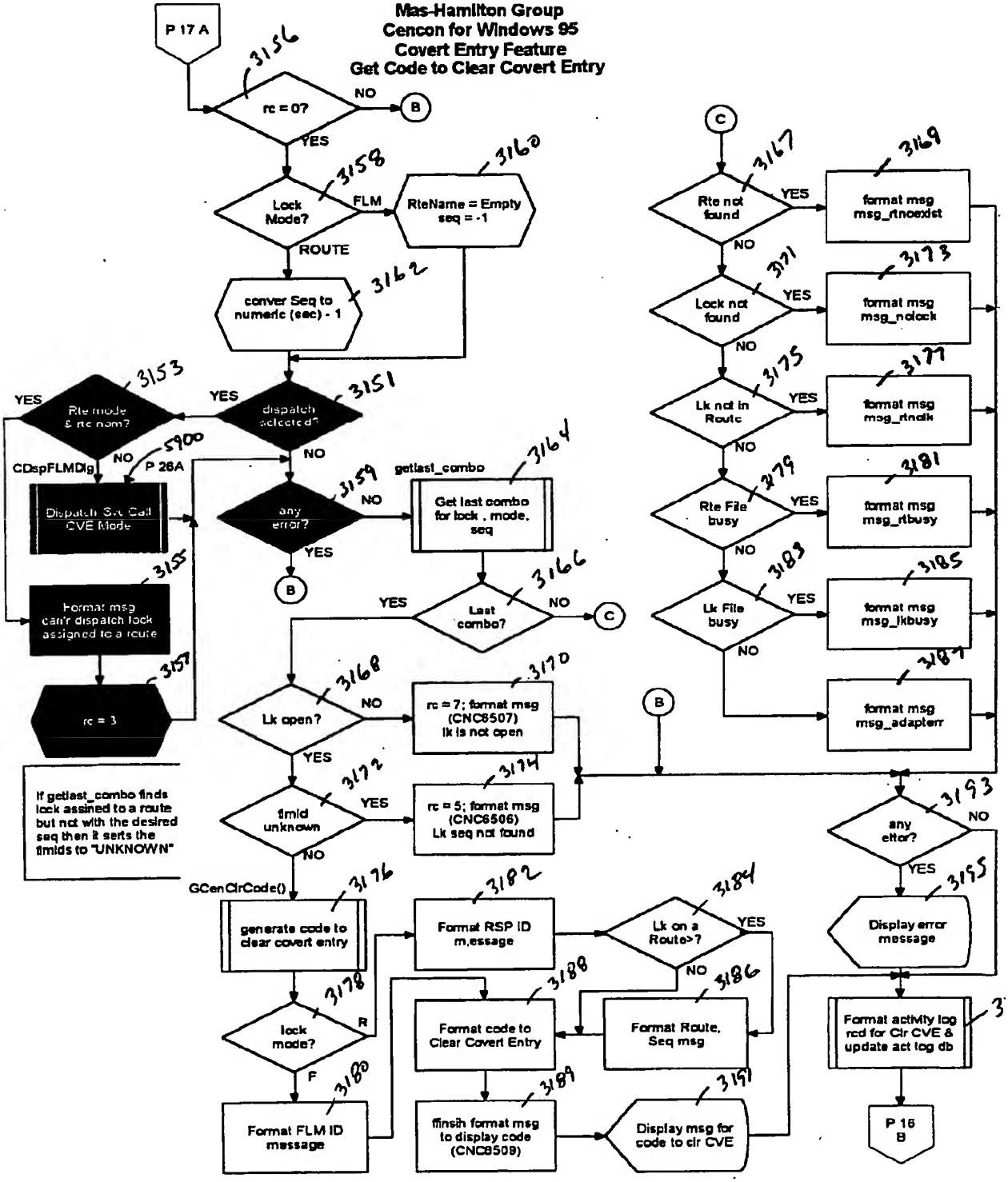

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
Get Code to Clear Covert Entry
P 17 A
3156
rc = 07?
NO
B
YES
3158
Lock Mode?
FLM
3160
RteName = Empty
seq = -1
ROUTE
3162
conver Seq to numeric (sec) - 1
3153
Rte mode & rte nam?
YES
YES
dispatch selected?
3151
NO
CDspFLMDlg
5900
P 28A
Dispatch Svc Call CVE Mode
3159
any error?
NO
getlast_combo
3164
Get last combo for lock , mode, seq
YES
B
3155
Format msg can't dispatch lock assigned to a route
3166
YES
Last combo?
NO
C
3157
rc = 3
3168
Lk open?
NO
3170
rc = 7; format msg (CNC8507) lk is not open
YES
If getlast_combo finds lock assined to a route but not with the desired seq then it sets the timids to "UNKNOWN"
3172
timid unknown
YES
3174
rc = 5; format msg (CNC8506) Lk seq not found
NO
GCenClrCode()
3176
generate code to clear covert entry
3182
Format RSP ID message
3184
Lk on a Route>?
YES
NO
3178
lock mode?
R
F
3188
Format code to Clear Covert Entry
3186
Format Route, Seq msg
3180
Format FLM ID message
3189
ffinsih format msg to display code (CNC8509)
3191
Display msg for code to clr CVE
C
3167
Rte not found
YES
3169
format msg msg_rtnoexist
NO
3171
Lock not found
YES
3173
format msg msg_nolock
NO
3175
Lk not in Route
YES
3177
format msg msg_rtnolk
NO
279
Rte File busy
YES
3181
format msg msg_rtbusy
NO
3183
Lk File busy
YES
3185
format msg msg_lkbusy
NO
3187
format msg msg_adapterr
B
3193
any error?
NO
YES
3195
Display error message
Format activity log rcd for Clr CVE & update act log db
3
P 16 B

FIG. 31 B-1

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
List Activity Log

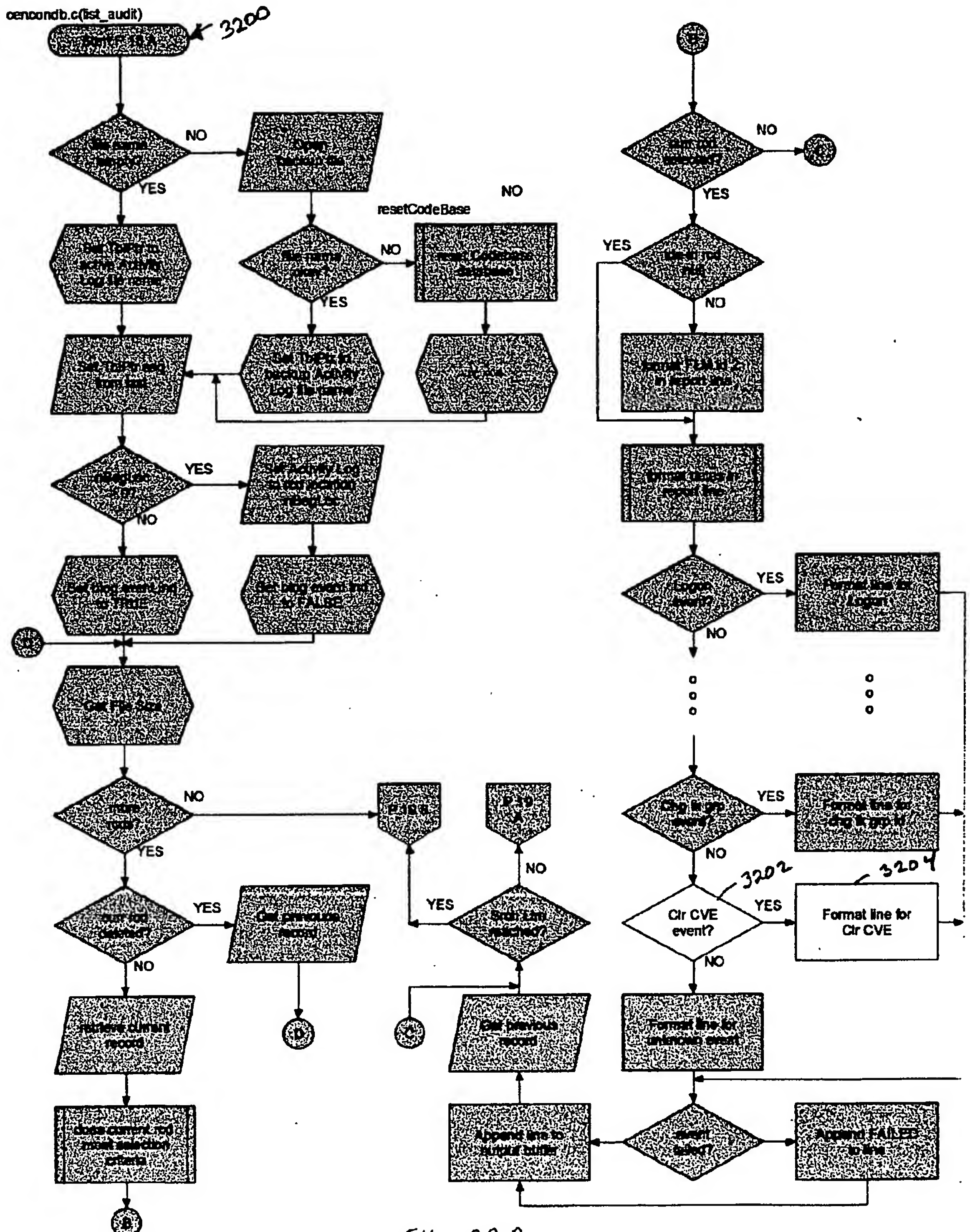


FIG. 32A

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
List Activity Log

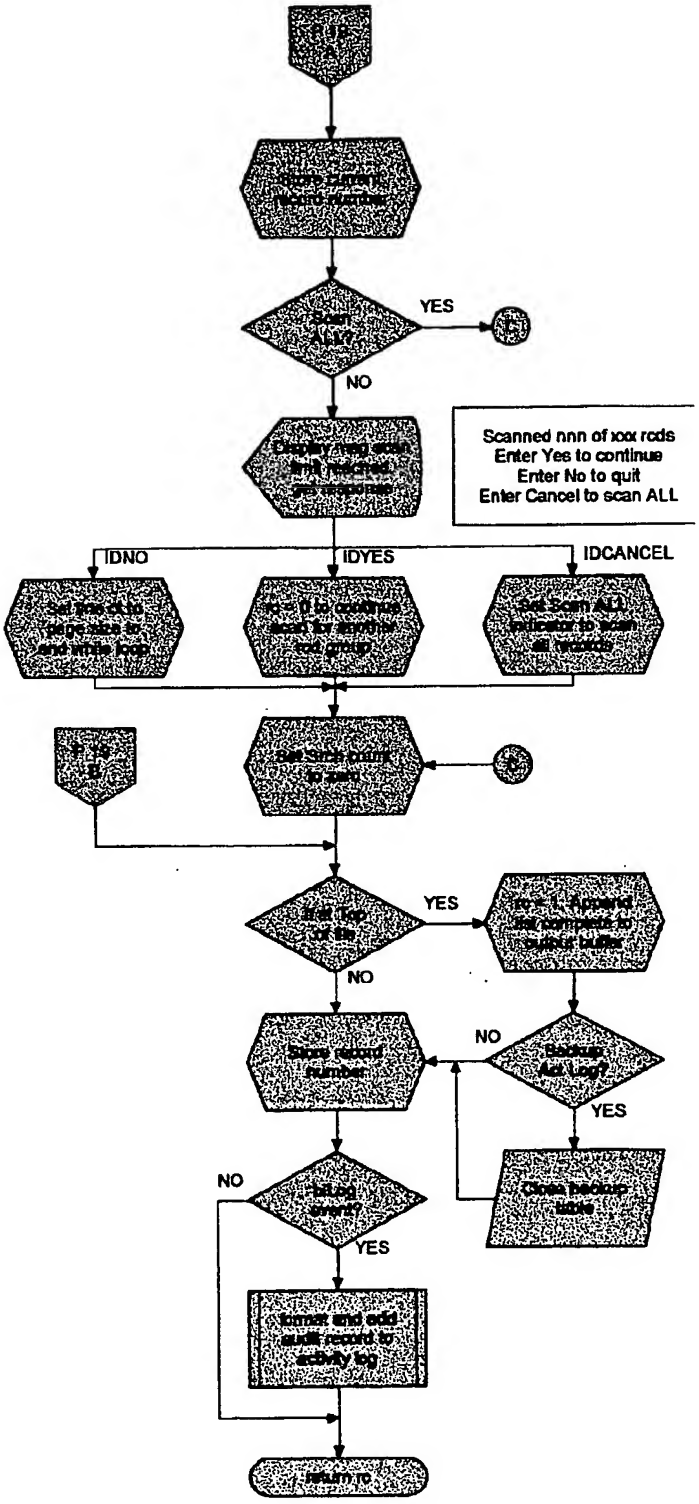
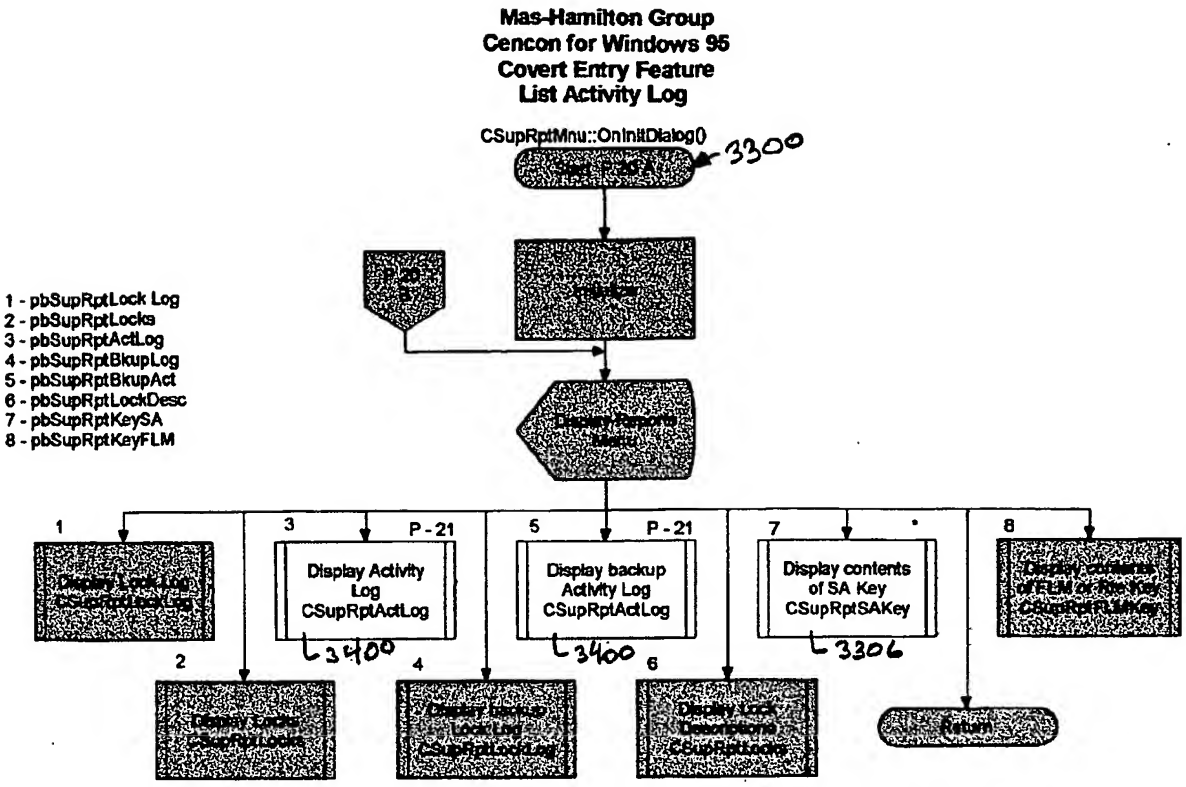


FIG. 32B



* Changes to tables in list_auditkey function in cenconbe.c to add "CE Cleared" to transdesc and "Covert Entry" to lockdesc. No changes to logic.

FIG. 33

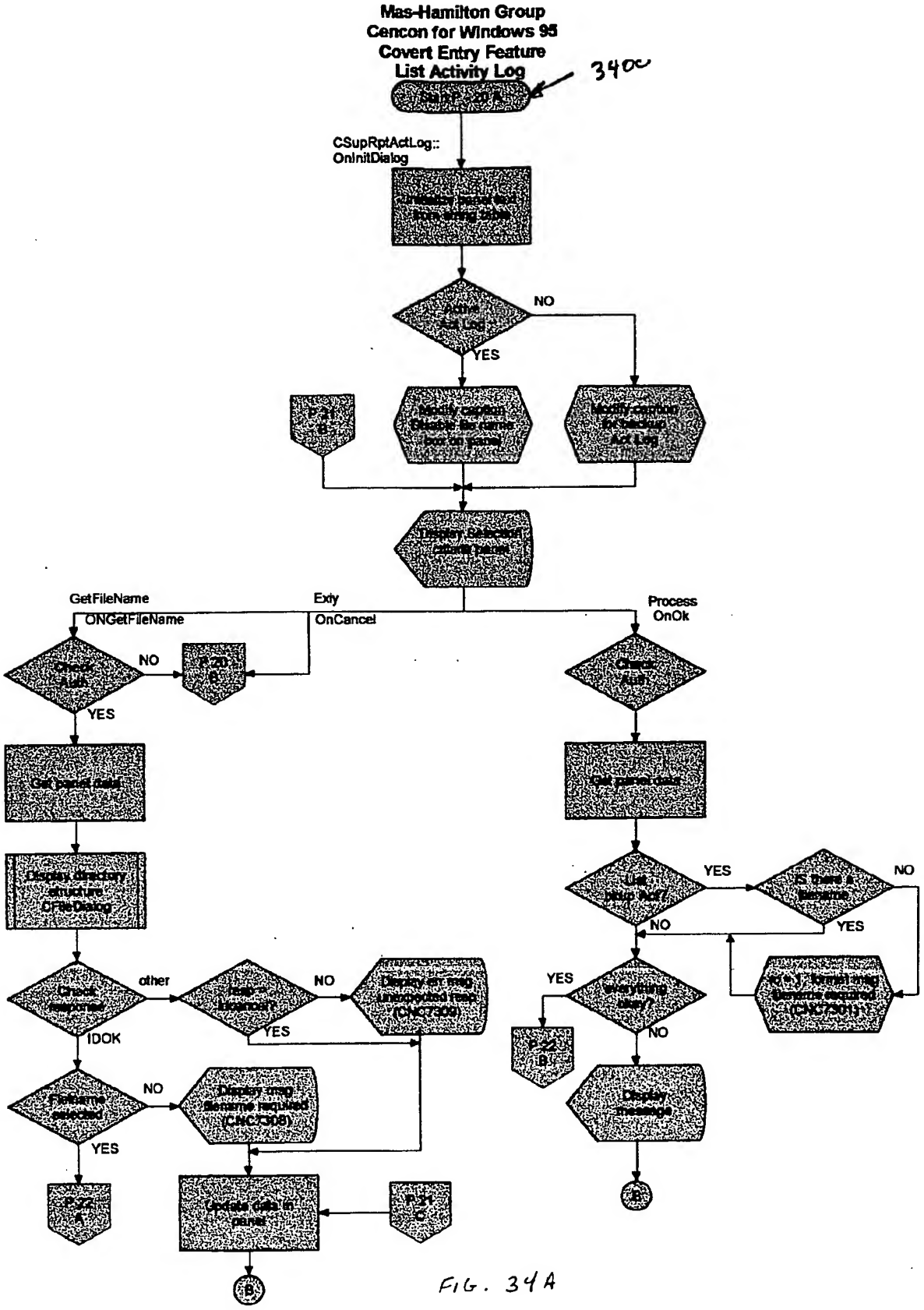

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
List Activity Log
340c
CSupRptActLog::
OnInitDialog
Active Act Log
NO
YES
Modify cencon Activate file name box on panel
Modify cencon to Deactivate Act Log
Display Selected cencon panel
GetFileName
ONGetFileName
Exty
OnCancel
Process
OnOk
Check Auth
NO
YES
Get panel data
Display directory structure CFileDialog
Check response
other
IDOK
NO
YES
Display error msg unsupported resp (CHC7309)
FileName selected
NO
YES
Display msg filename required (CHC7308)
Update data in panel
Use Help Act?
YES
NO
Is there a filename
NO
YES
Display filename required (CHC7301)
Everything okay?
YES
NO
Display message

FIG. 34A

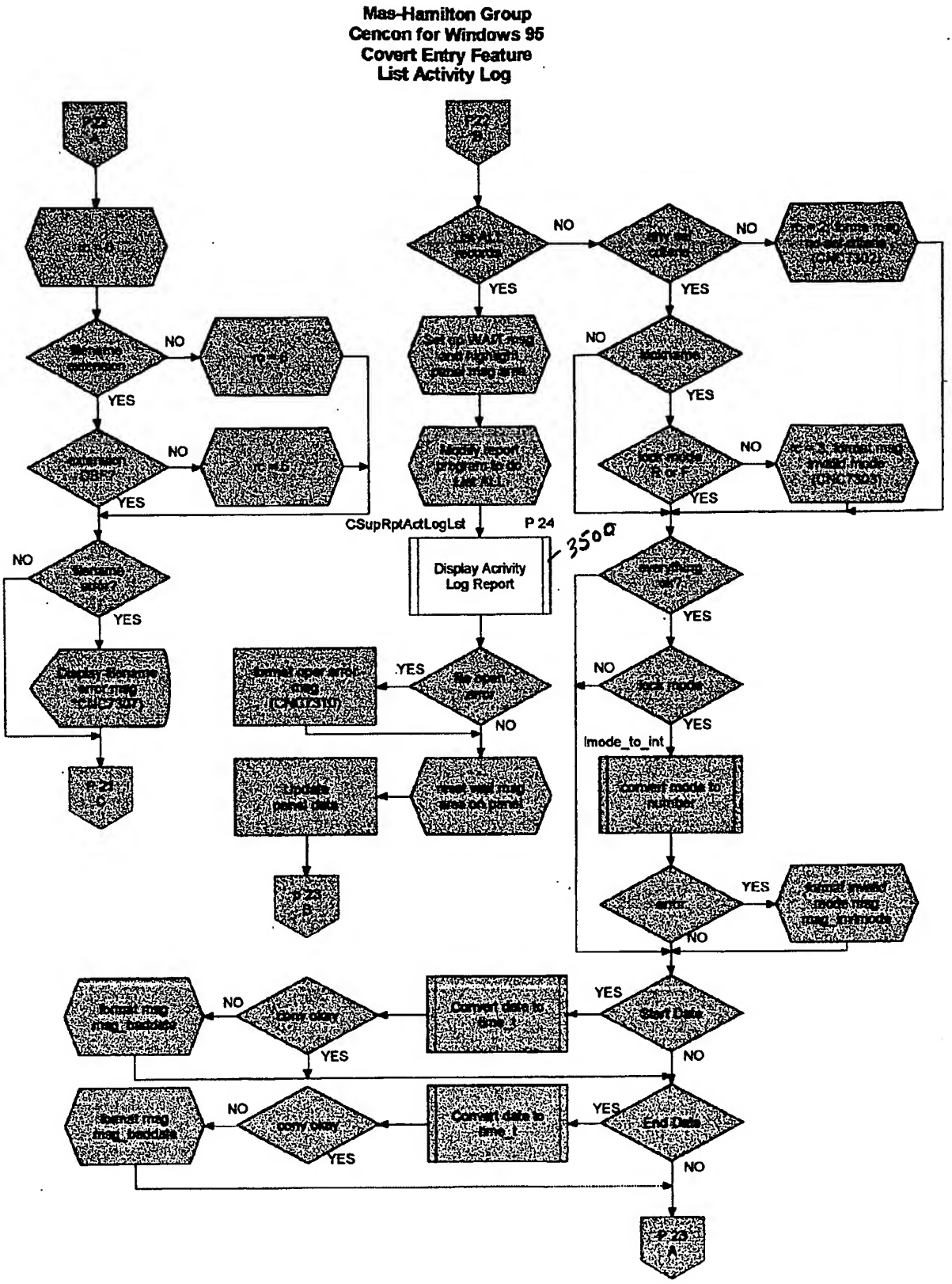

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
List Activity Log
YES
NO
CSupRptActLogLst
P 24
Display Acrivity Log Report
350a
Imode_to_int

FIG. 34B

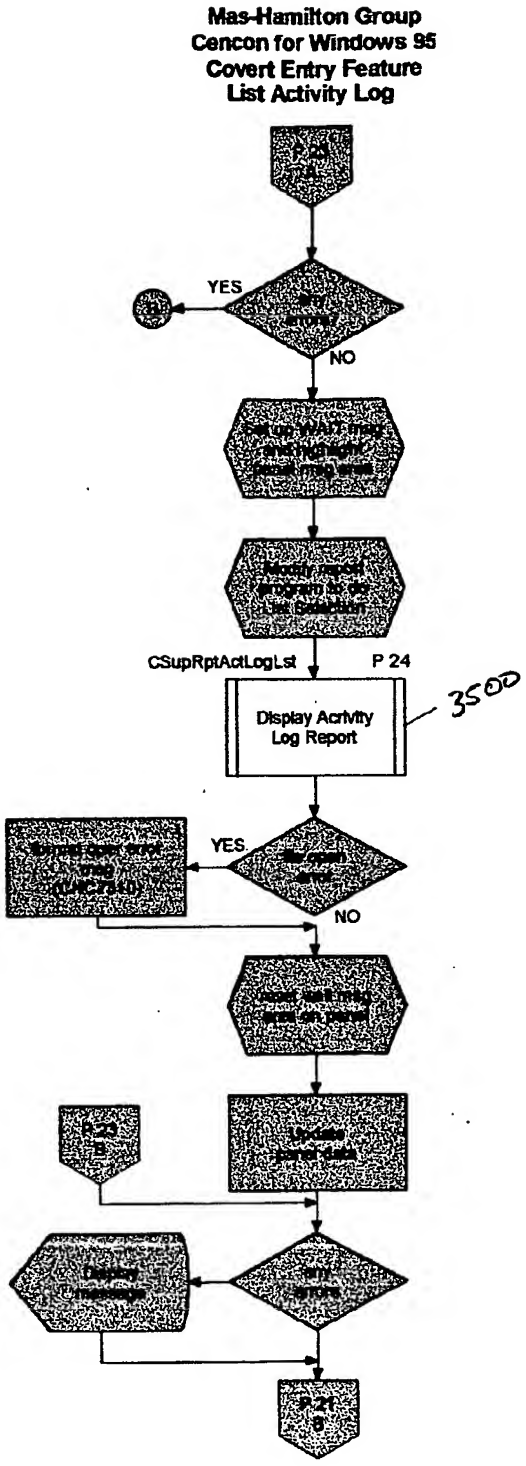

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
List Activity Log
YES
NO
CSupRptActLogLst
P 24
Display Activity Log Report
3500
YES
NO

FIG. 34C

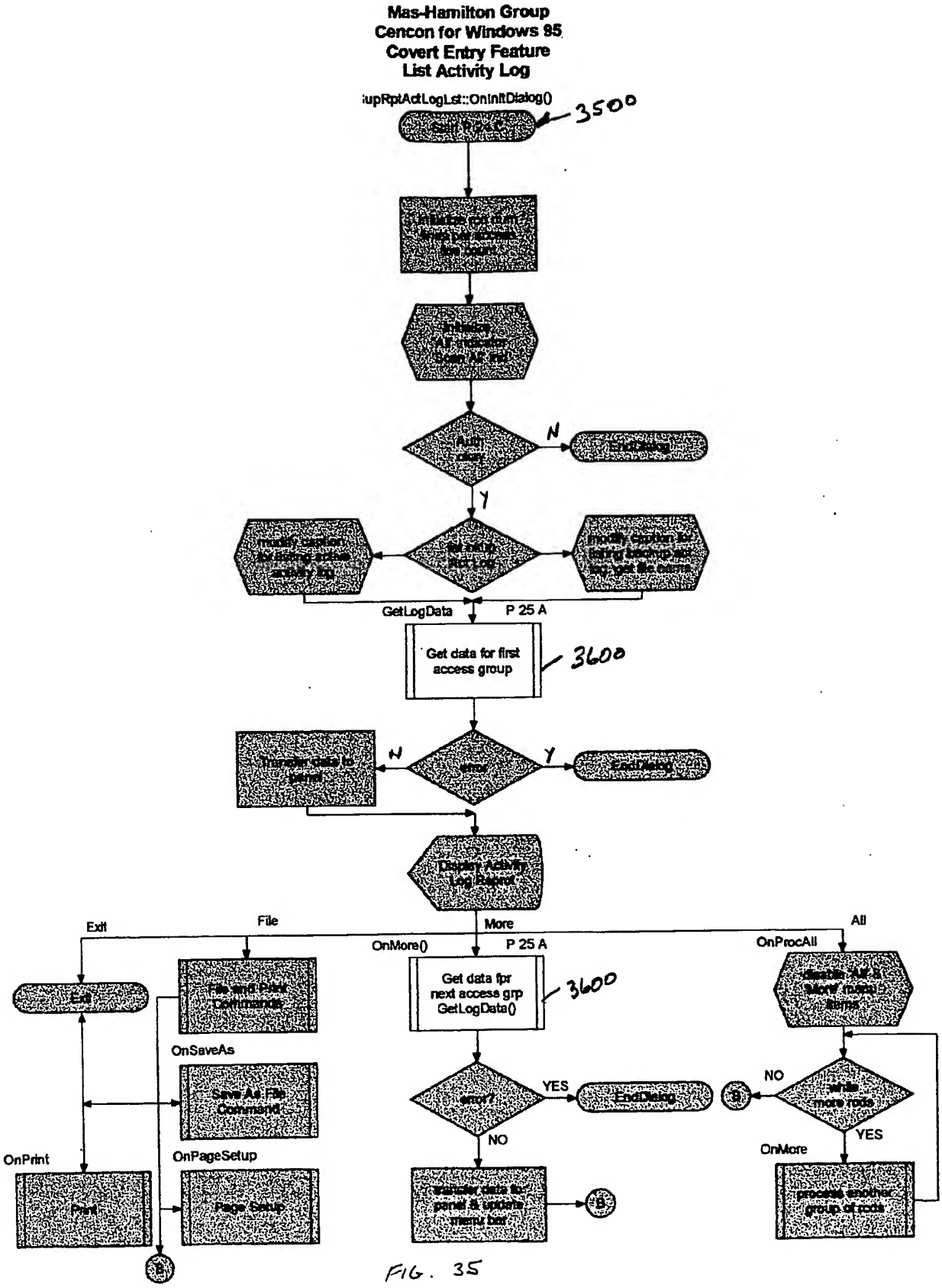

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
List Activity Log
3500
N
Y
GetLogData
P 25 A
Get data for first access group
3600
Exit
File
More
All
OnMore()
P 25 A
OnProcAll
Get data for next access grp GetLogData()
3600
OnSaveAs
YES
NO
OnPrint
OnPageSetup
OnMore
YES
NO

FIG. 35

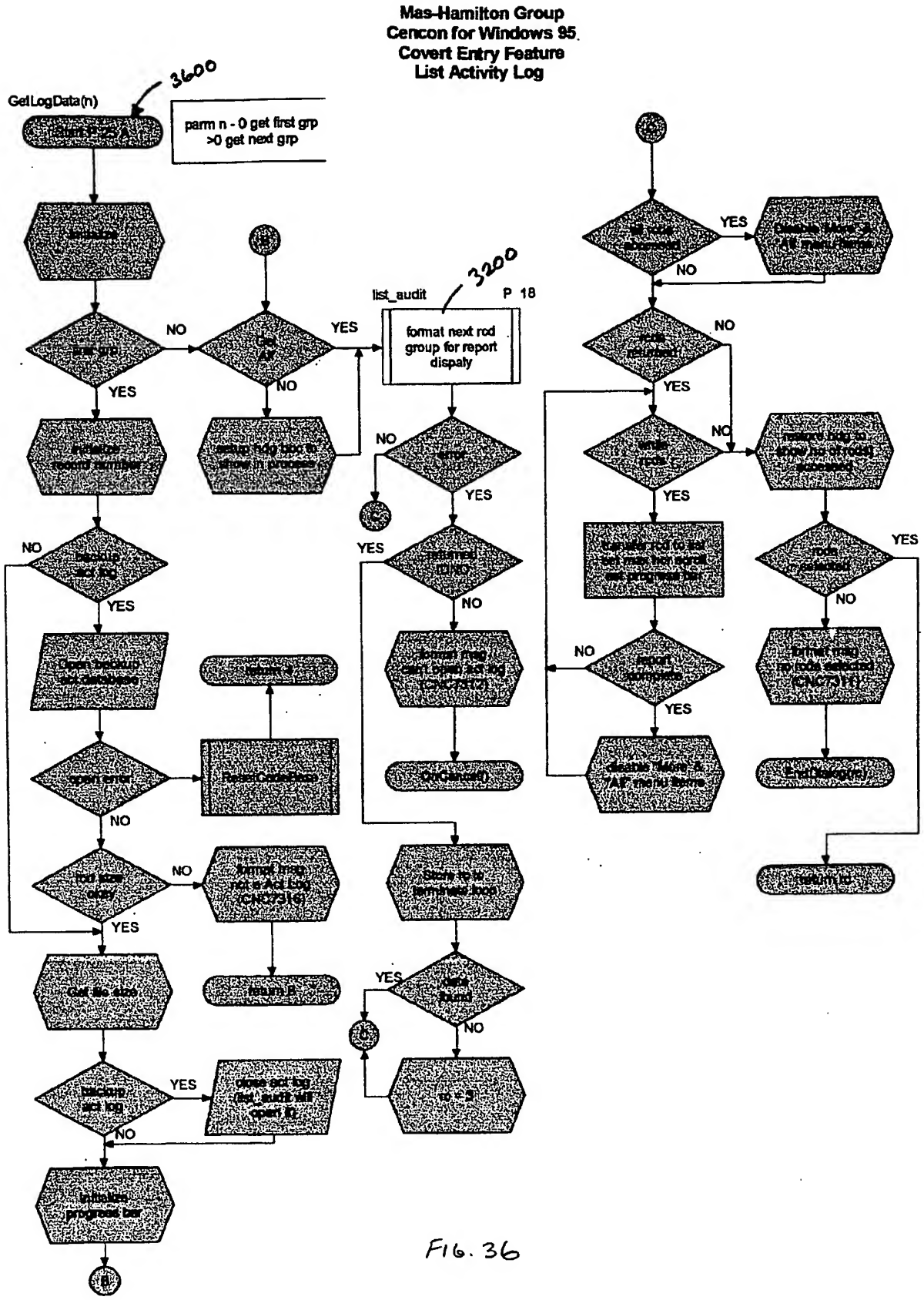

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
List Activity Log
3600
GetLogData(n)
parm n - 0 get first grp
>0 get next grp
YES
NO
3200
list_audit
P 18
format next rod group for report dispaly

Fig. 36

Cencon 2000 for Windows 95
Covert Entry (Cover) Close Code
Flowcharts

Introduction

A special option has been added to the Mas-Hamilton Cencon 2000 ATM/Lock with the Covert Entry Feature. The lock senses removal of the back cover and records the event in its memory. The next time the lock is opened it modifies the Close Seal to indicate that the back cover had been removed. Modifications are required in the Cencon 2000 for Window 95 software to recognize the modified Close Seal and to allow it to be used to close the service call. The following flowcharts show the logic flow of the modifications requires to support this option. Modified or added logic is indicated by grey figures. Existing unmodified figures required to show where the modifications occur within the application logic are shown as white.

Flowchart Index

3700

FIG. 37

Cencon 2000 for Windows 95
Covert Entry (Cover) Close Code
Flowcharts

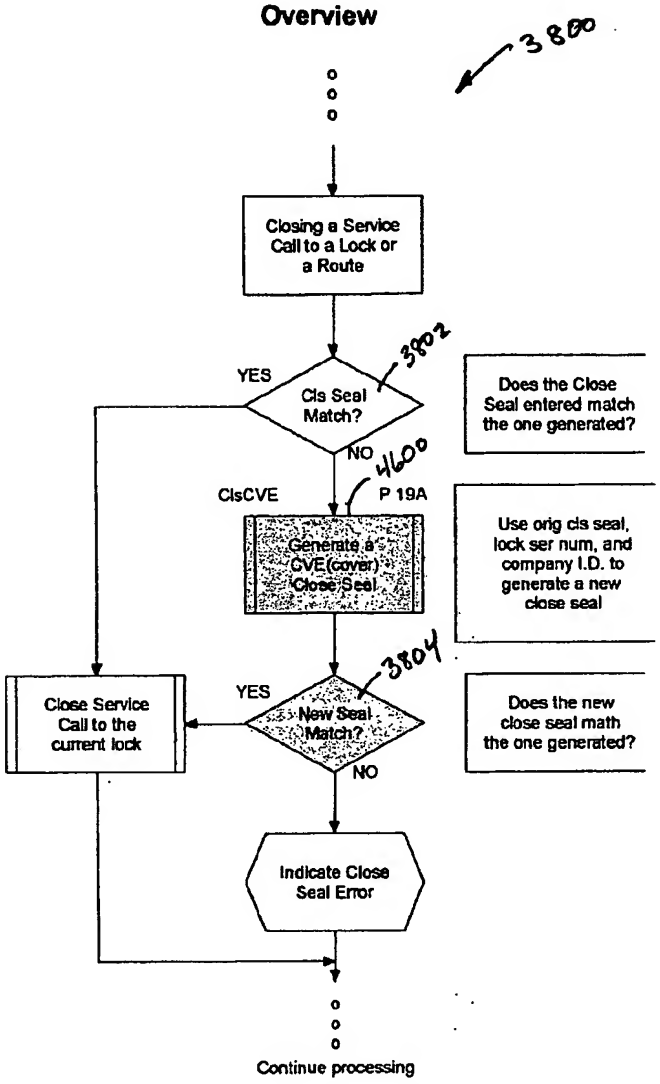


FIG. 38

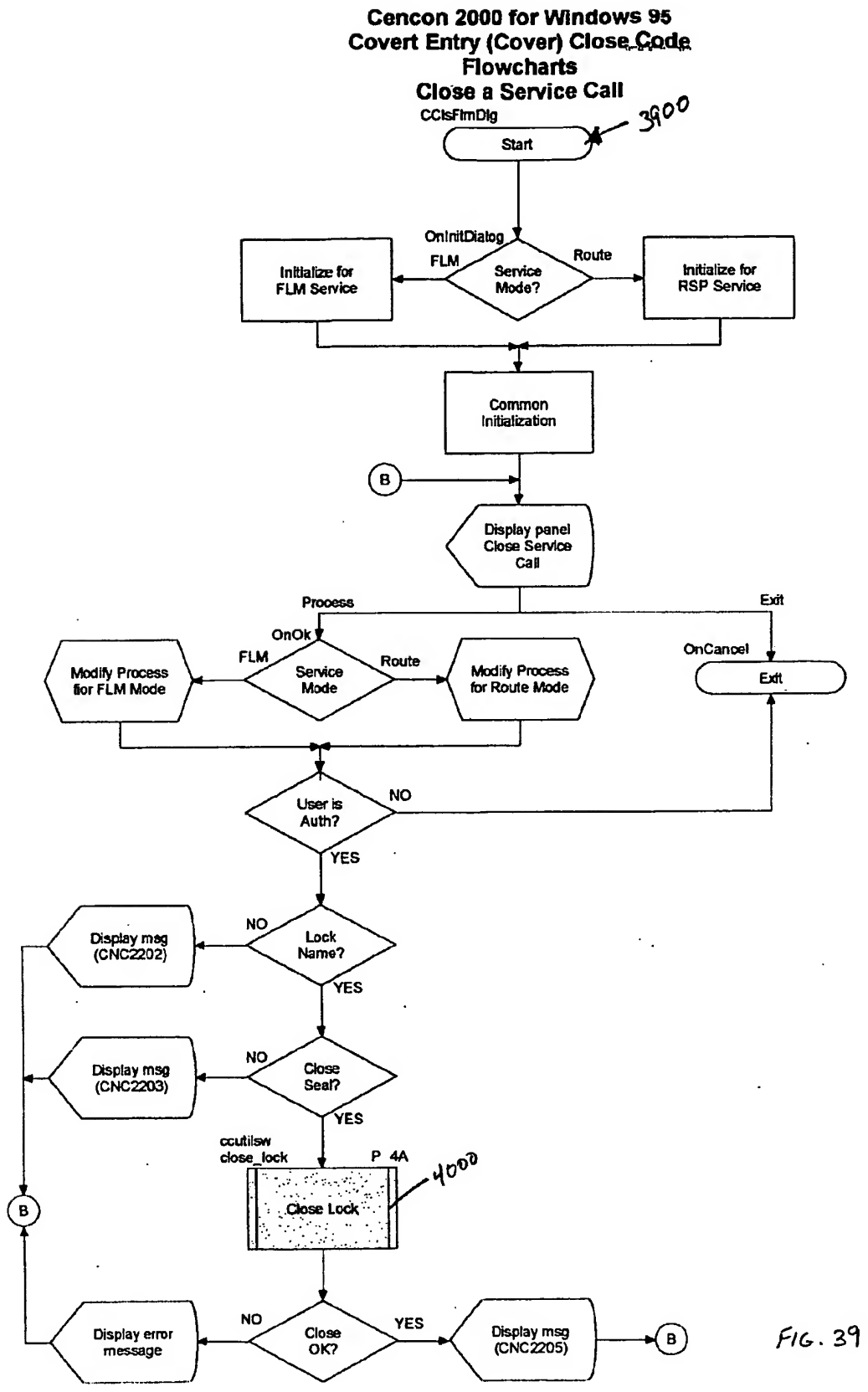

Cencon 2000 for Windows 95
Covert Entry (Cover) Close Code
Flowcharts
Close a Service Call
CCIsFlmDlg
3900
Start
OnInitDialog
FLM
Service Mode?
Route
Initialize for FLM Service
Initialize for RSP Service
Common Initialization
B
Display panel Close Service Call
Process
Exit
OnOk
FLM
Service Mode
Route
Modify Process for FLM Mode
Modify Process for Route Mode
OnCancel
Exit
User is Auth?
NO
YES
Display msg (CNC2202)
NO
Lock Name?
YES
Display msg (CNC2203)
NO
Close Seal?
YES
ccutilsw
close_lock
P 4A
4000
Close Lock
B
Display error message
NO
Close OK?
YES
Display msg (CNC2205)
B

FIG. 39

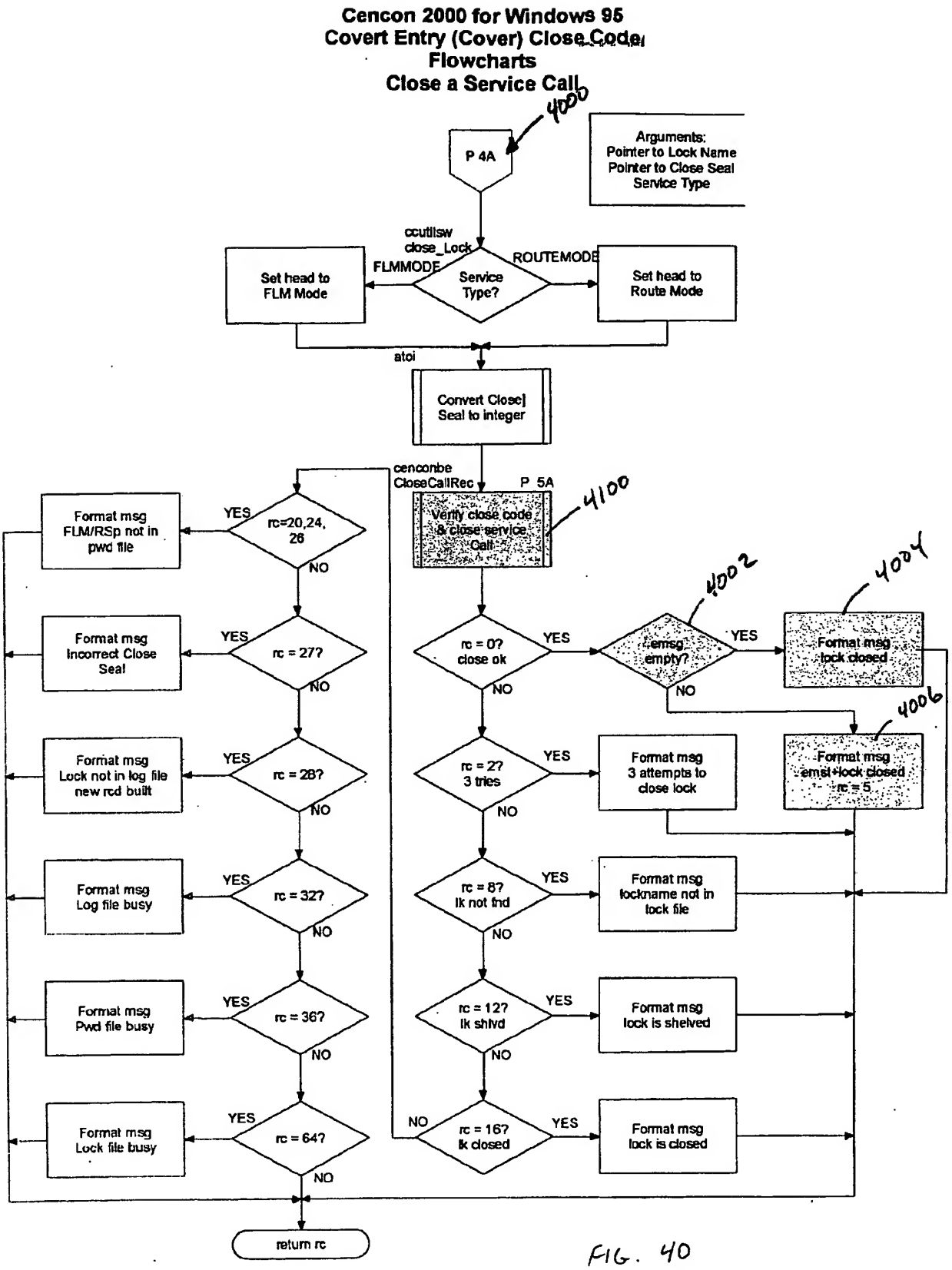

Cencon 2000 for Windows 95
Covert Entry (Cover) Close Code
Flowcharts
Close a Service Call
4000
P 4A
Arguments:
Pointer to Lock Name
Pointer to Close Seal
Service Type
ccutilsw
close_Lock
FLMMODE
Service Type?
ROUTEMODE
Set head to FLM Mode
Set head to Route Mode
atoi
Convert Close] Seal to integer
cenconbe
CloseCallRec
P 5A
4100
Verify close code & close service Call
4002
4004
4006
rc = 0? close ok
YES
NO
emsg empty?
YES
NO
Format msg lock closed
Format msg emsl lock closed rc = 5
rc = 2? 3 tries
YES
NO
Format msg 3 attempts to close lock
rc = 8? lk not fnd
YES
NO
Format msg lockname not in lock file
rc = 12? lk shlvd
YES
NO
Format msg lock is shelved
rc = 16? lk closed
NO
YES
Format msg lock is closed
rc=20,24, 26
YES
NO
Format msg FLM/RSp not in pwd file
rc = 27?
YES
NO
Format msg Incorrect Close Seal
rc = 28?
YES
NO
Format msg Lock not in log file new rcd built
rc = 32?
YES
NO
Format msg Log file busy
rc = 36?
YES
NO
Format msg Pwd file busy
rc = 64?
YES
NO
Format msg Lock file busy
return rc

FIG. 40

Cencon 2000 for Windows 95
Covert Entry (Cover) Close Code .
Flowcharts
Close a Service Call

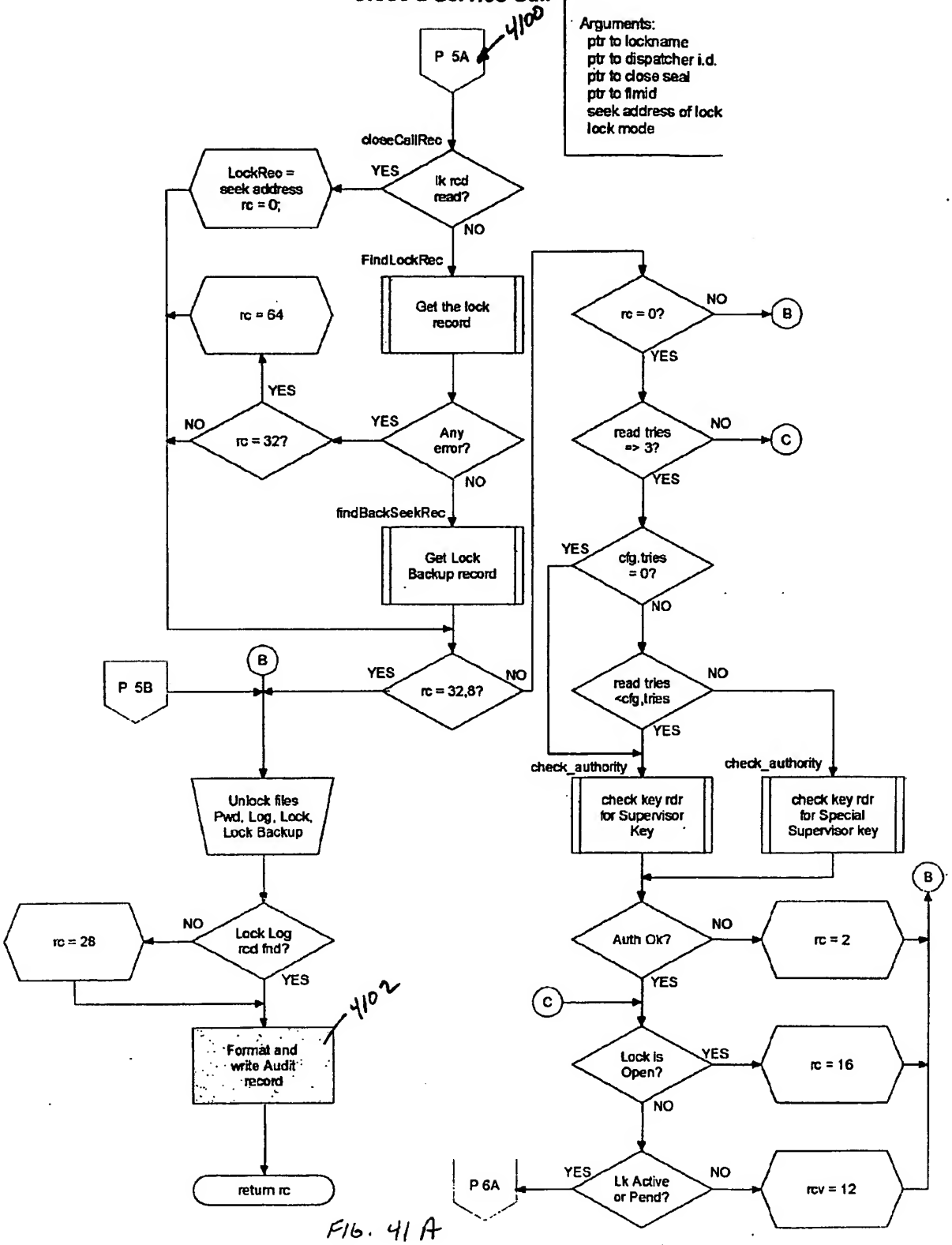


FIG. 41A

Cencon 2000 for Windows 95
Covert Entry (Cover) Close Code
Flowcharts
Close a Service Call

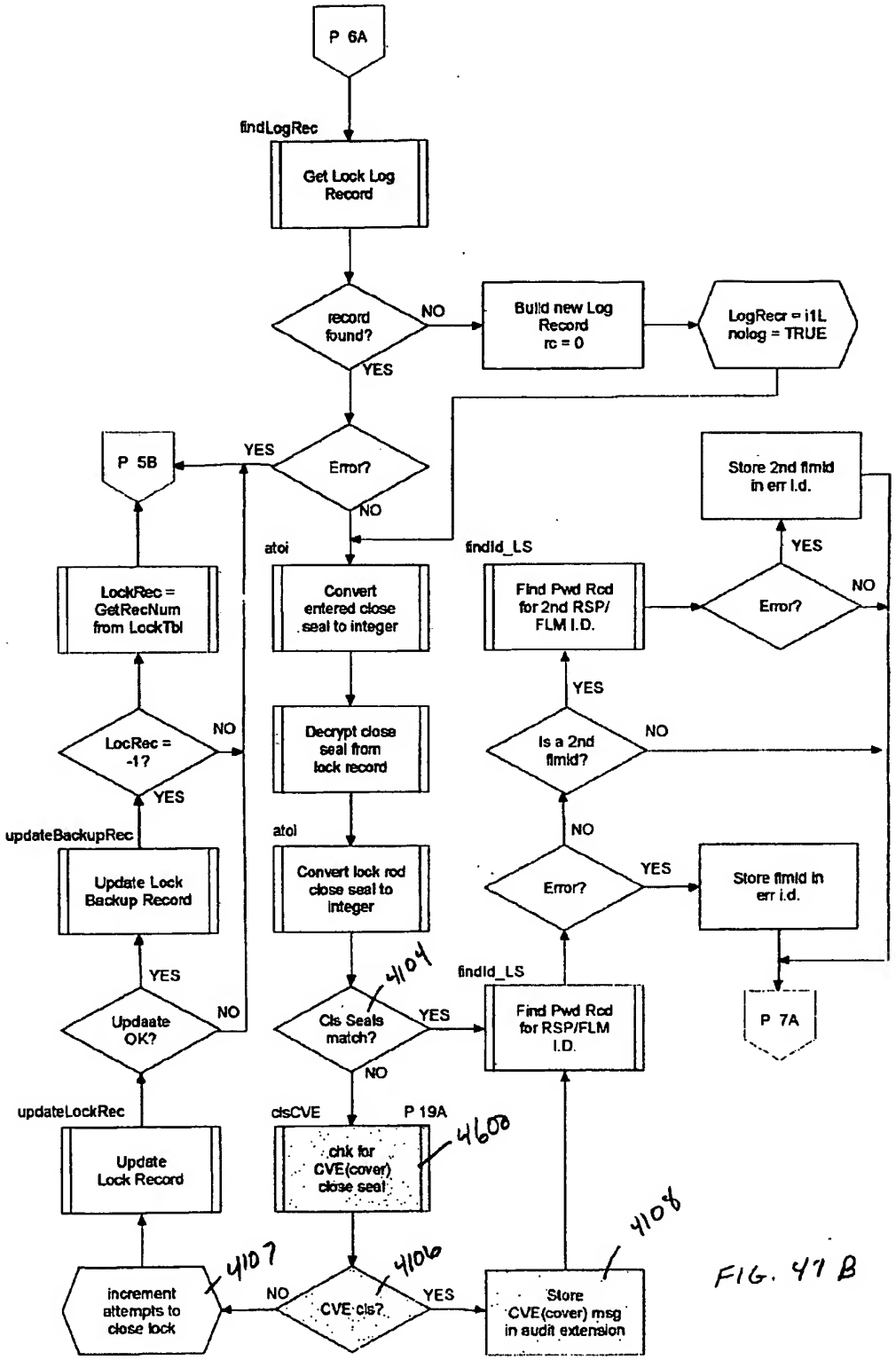


FIG. 41B

Cencon 2000 for Windows 95
Covert Entry (Cover) Close Code.
Flowcharts
Close a Service Call

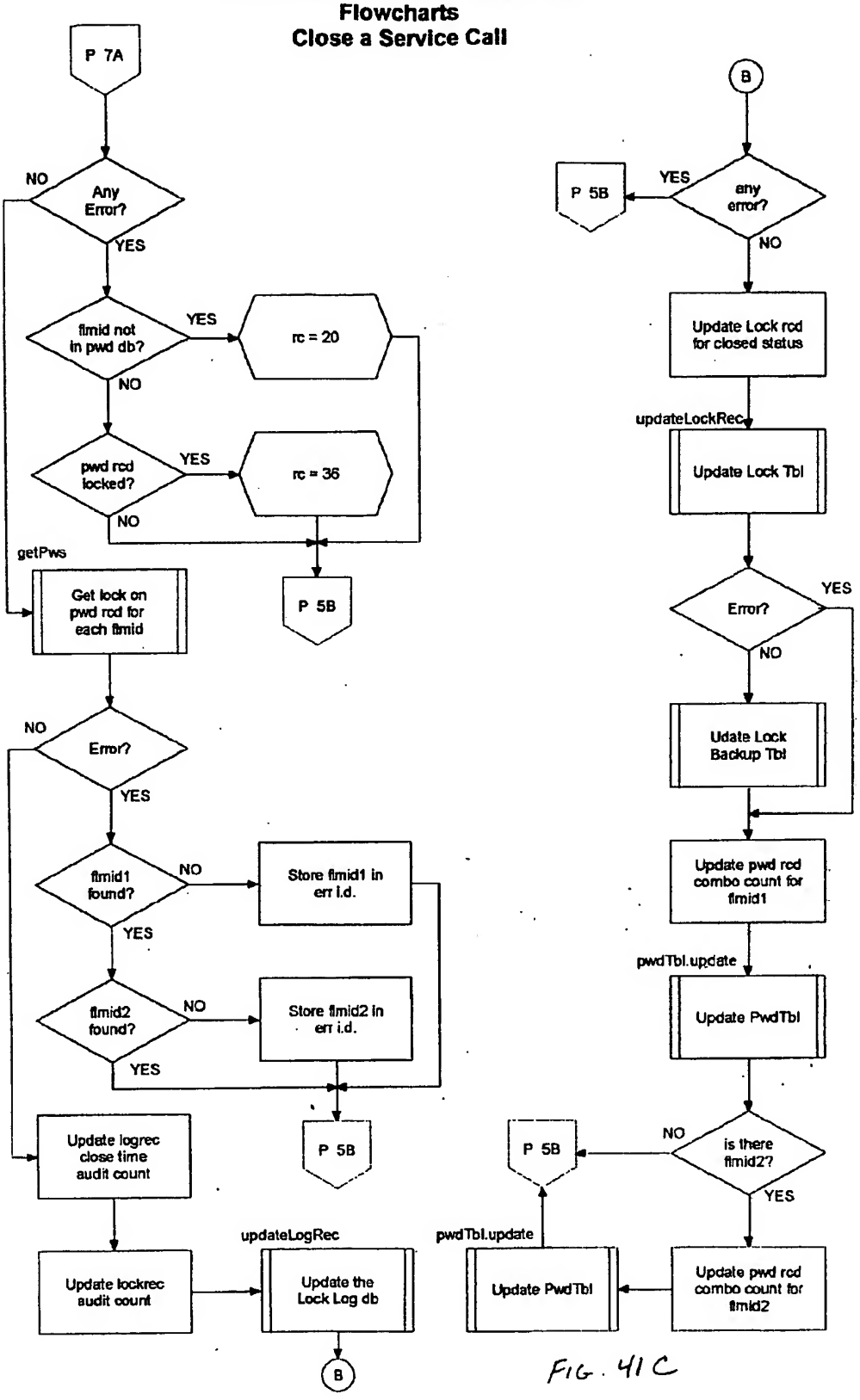


FIG. 41C

Cencon 2000 for Windows 95
Covert Entry (Cover) Close Code
Flowcharts
Close a Route Using a Menu

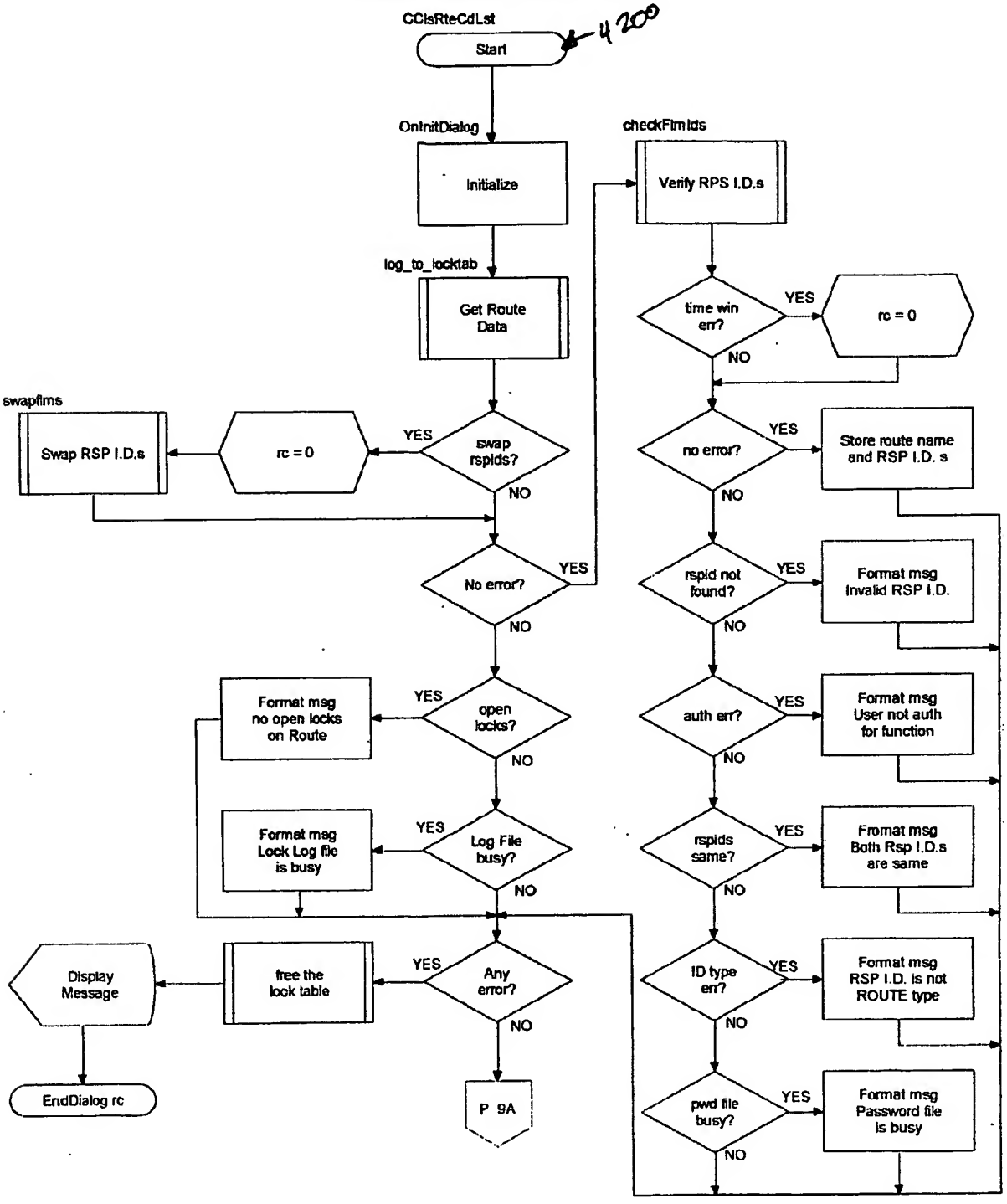


FIG. 42A

Cencon 2000 for Windows 95
Covert Entry (Cover) Close Code
Flowcharts
Close a Route Using a Menu

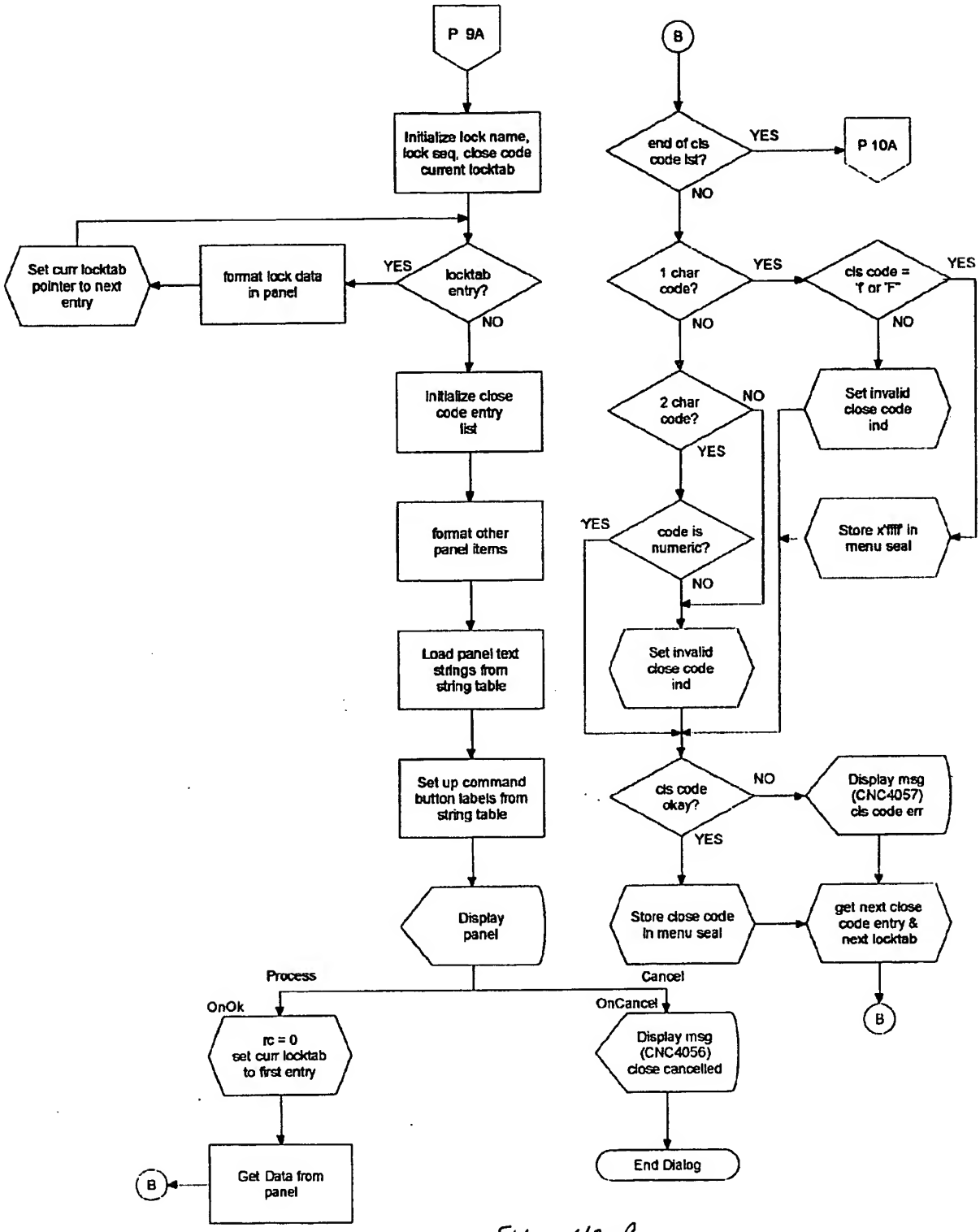


FIG. 42B

Cencon 2000 for Windows 95
Covert Entry (Cover) Close Code
Flowcharts
Close a Route Using a Menu

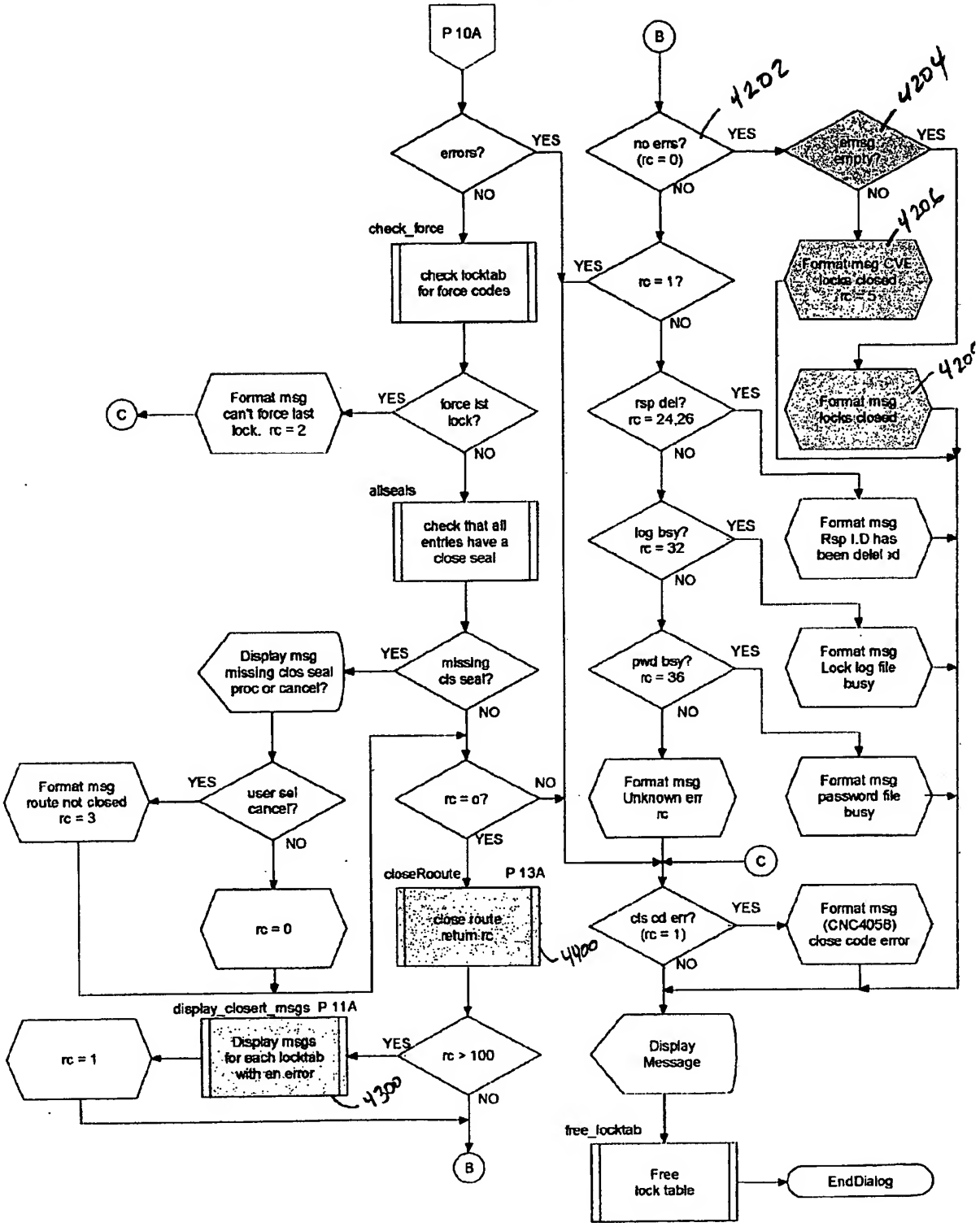


FIG 42C

Cencon 2000 for Windows 95
Covert Entry (Cover) Close Code
Flowcharts
Close a Route Using a Menu

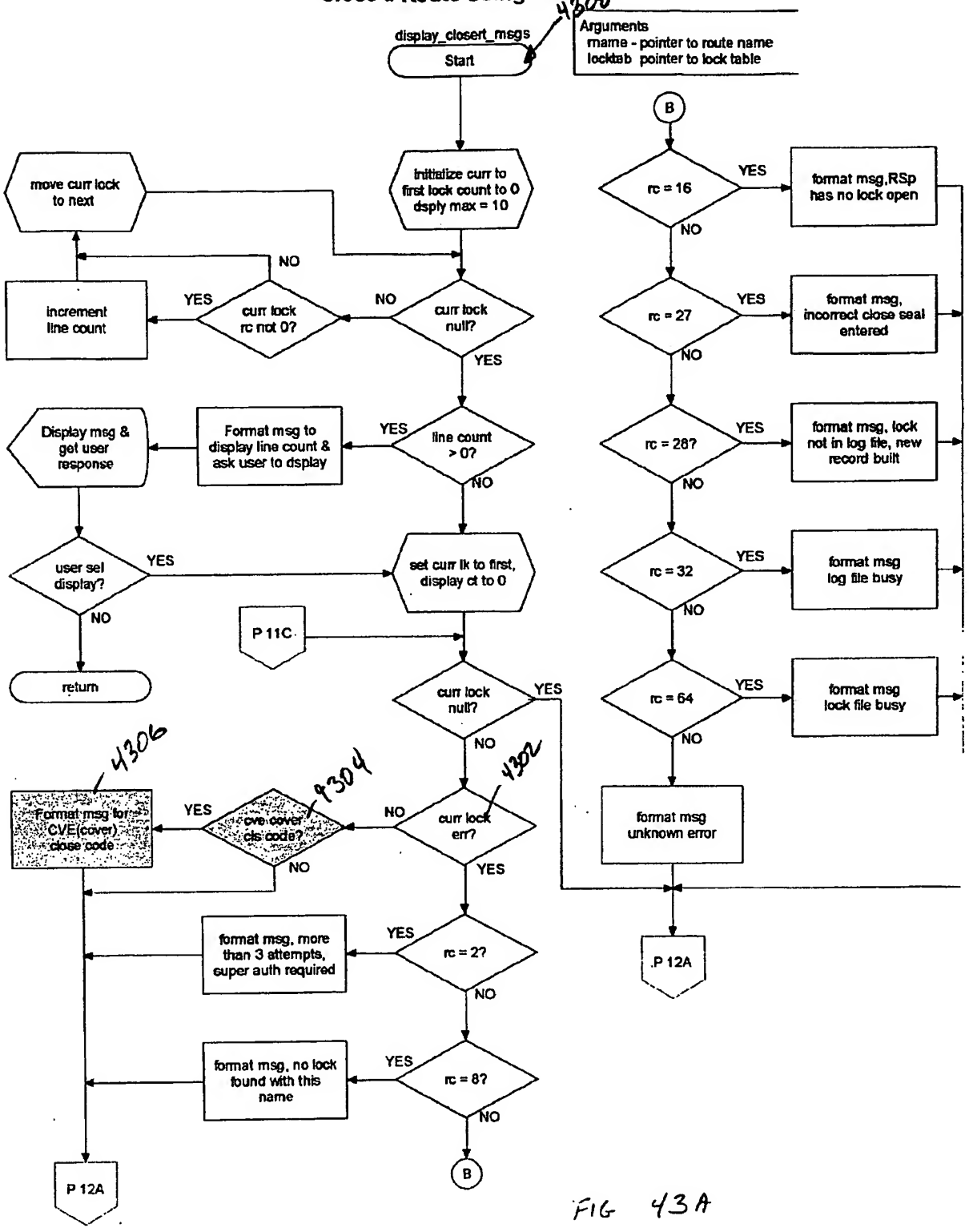


FIG 43A

Cencon 2000 for Windows 95
Covert Entry (Cover) Close Code
Flowcharts
Close a Route Using a Menu

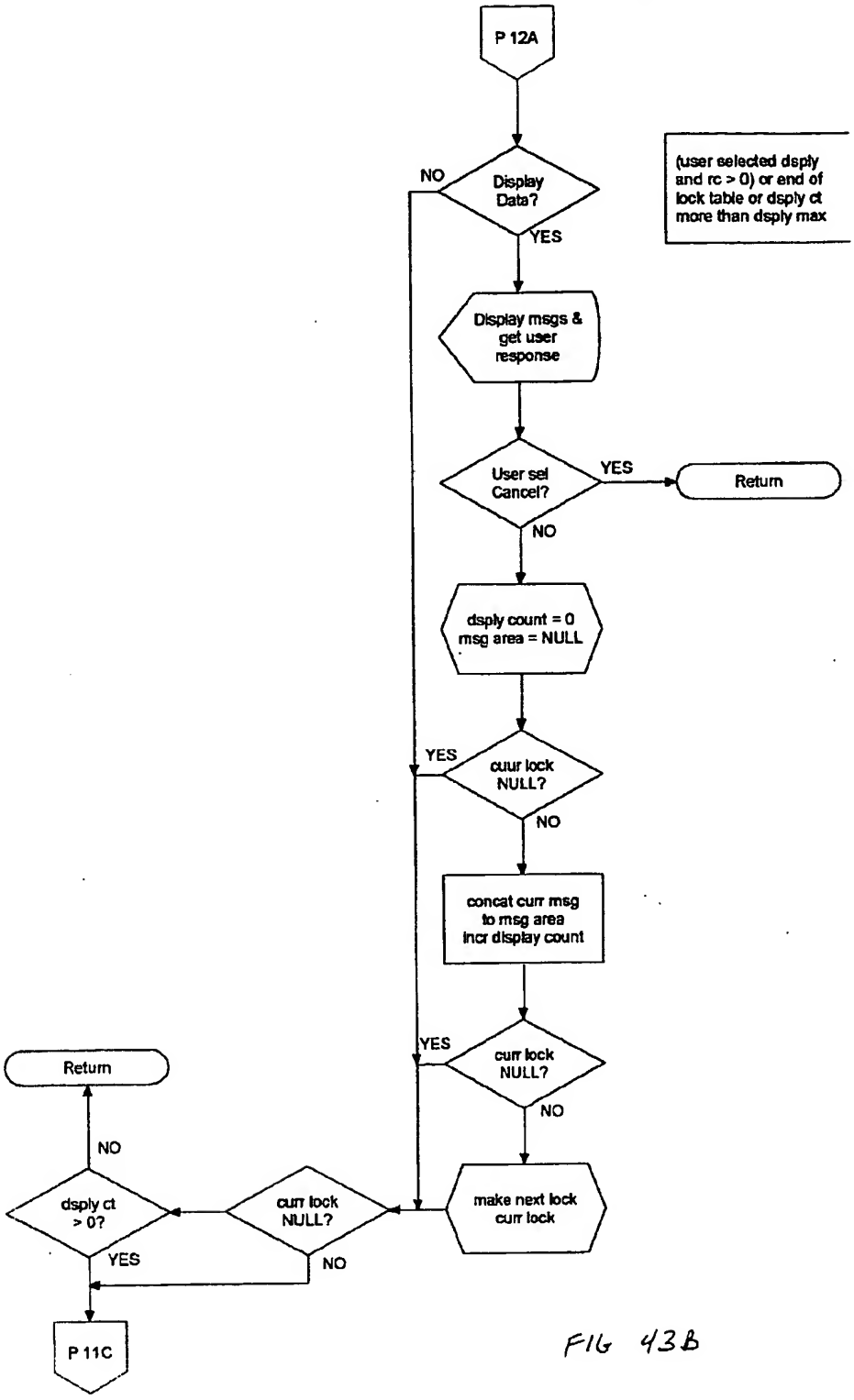


FIG 43B

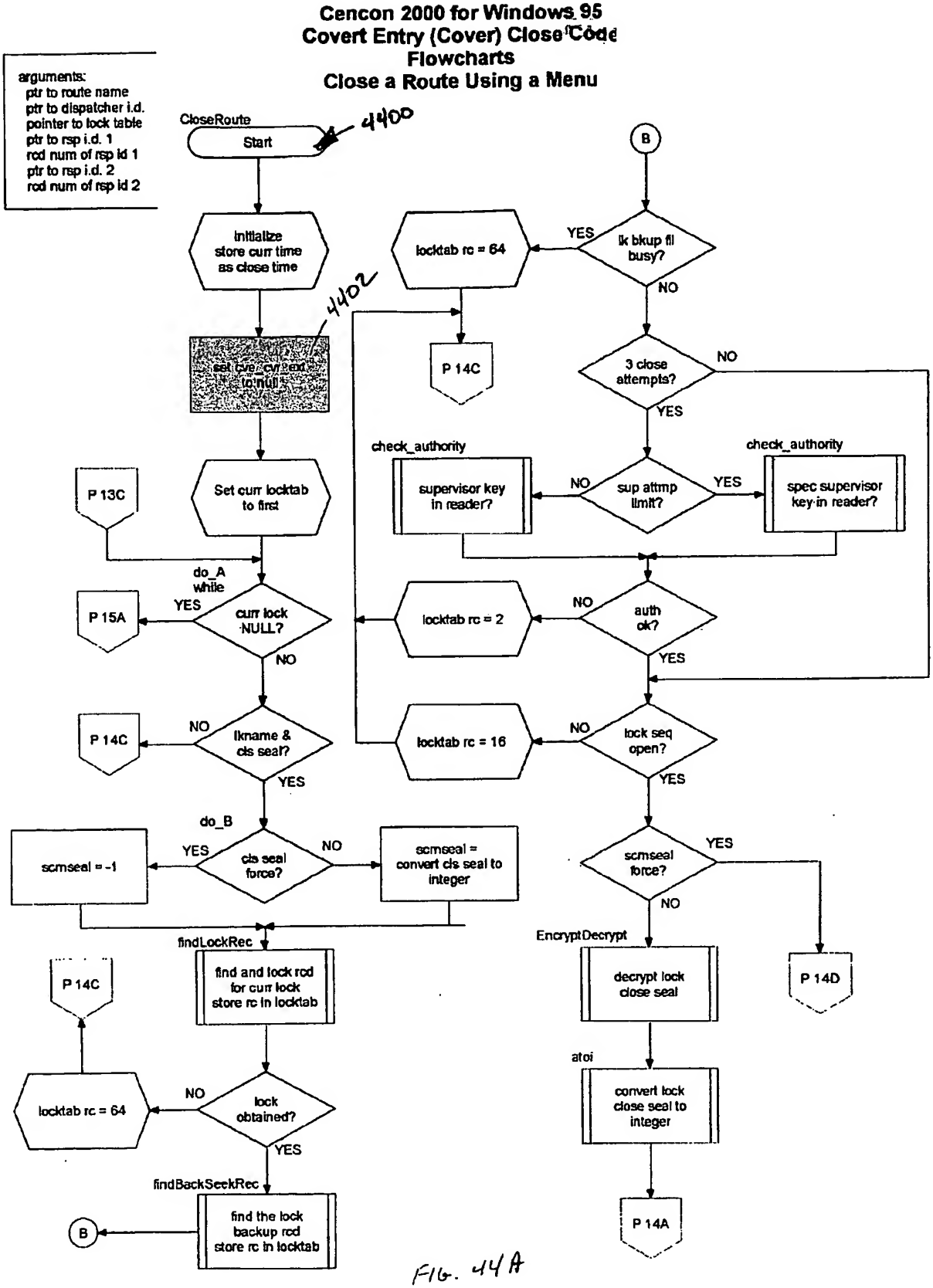

Cencon 2000 for Windows 95
Covert Entry (Cover) Close Code
Flowcharts
Close a Route Using a Menu
arguments:
ptr to route name
ptr to dispatcher i.d.
pointer to lock table
ptr to rsp i.d. 1
rcd num of rsp id 1
ptr to rsp i.d. 2
rcd num of rsp id 2
CloseRoute
Start
4400
initialize
store curr time
as close time
4402
set cve_cvr_ent
to null
Set curr locktab
to first
P 13C
do_A
while
curr lock
NULL?
YES
P 15A
NO
lkname &
cls seal?
NO
P 14C
YES
do_B
cls seal
force?
YES
scmseal = -1
NO
scmseal =
convert cls seal to
integer
findLockRec
find and lock rcd
for curr lock
store rc in locktab
lock
obtained?
NO
locktab rc = 64
YES
P 14C
findBackSeekRec
find the lock
backup rcd
store rc in locktab
B
B
lk bkup fil
busy?
YES
locktab rc = 64
NO
P 14C
3 close
attempts?
NO
YES
check_authority
sup attmp
limit?
NO
supervisor key
in reader?
YES
check_authority
spec supervisor
key in reader?
auth
ok?
NO
locktab rc = 2
YES
lock seq
open?
NO
locktab rc = 16
YES
scmseal
force?
YES
NO
EncryptDecrypt
decrypt lock
close seal
P 14D
atoi
convert lock
close seal to
integer
P 14A

Fig. 44A

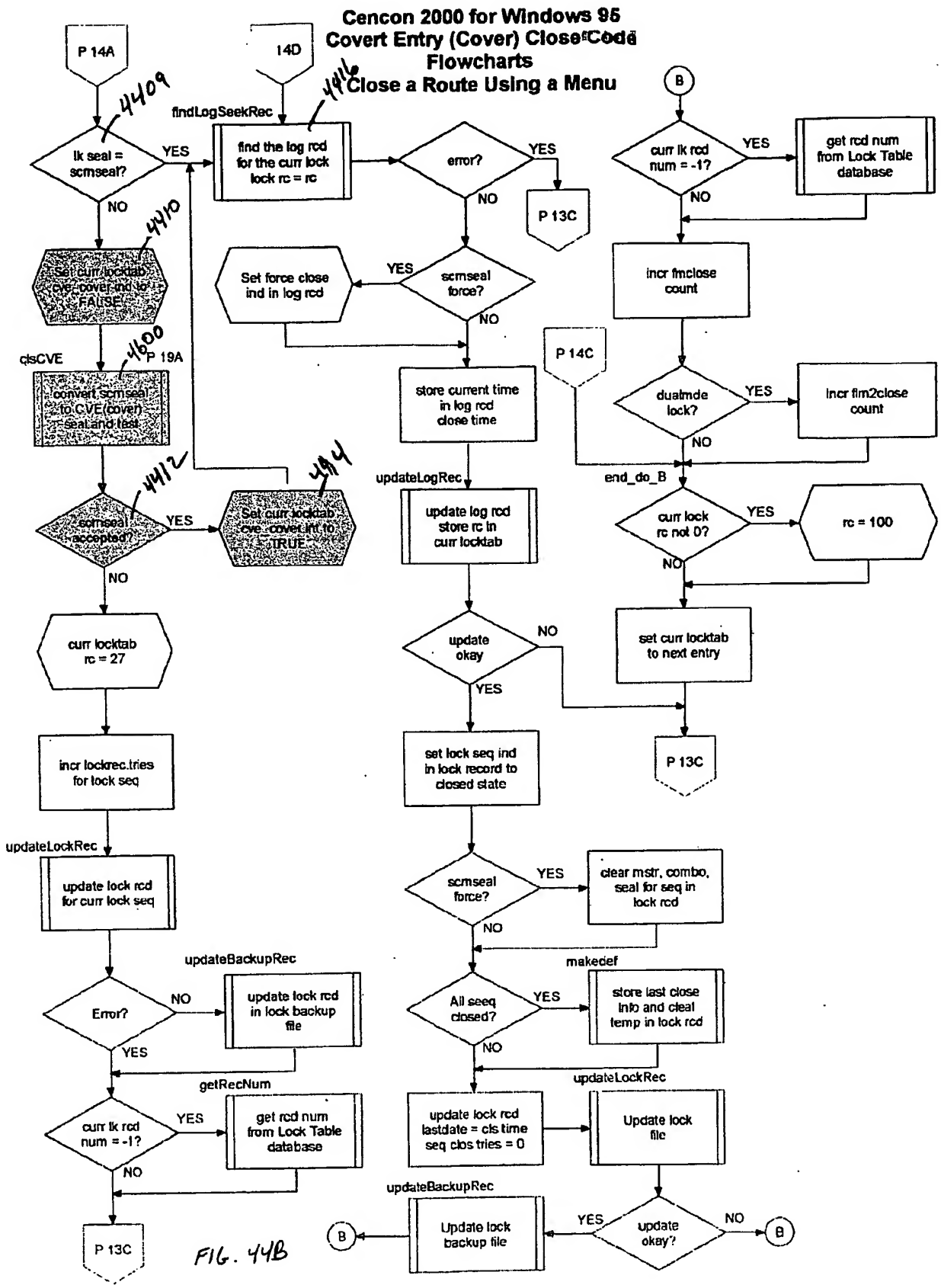

Cencon 2000 for Windows 95
Covert Entry (Cover) Close Code
Flowcharts
Close a Route Using a Menu
P 14A
14D
4409
4416
findLogSeekRec
lk seal = scmseal?
YES
NO
find the log rcd for the curr lock lock rc = rc
error?
YES
NO
P 13C
B
curr lk rcd num = -1?
YES
NO
get rcd num from Lock Table database
4410
Set curr locktab cve-cover-ind to FALSE
Set force close ind in log rcd
YES
scmseal force?
NO
incr fmclose count
clsCVE
4600
P 19A
convert scmseal to CVE(cover) seal and last
store current time in log rcd close time
P 14C
dualmde lock?
YES
NO
incr fim2close count
4412
4414
scmseal accepted?
YES
Set curr locktab cve-cover-ind to TRUE
updateLogRec
update log rcd store rc in curr locktab
end_do_B
curr lock rc not 0?
YES
NO
rc = 100
curr locktab rc = 27
update okay
NO
YES
set curr locktab to next entry
incr lockrec.tries for lock seq
set lock seq ind in lock record to closed state
P 13C
updateLockRec
update lock rcd for curr lock seq
scmseal force?
YES
NO
clear mstr, combo, seal for seq in lock rcd
updateBackupRec
Error?
NO
YES
update lock rcd in lock backup file
makedef
All seeq closed?
YES
NO
store last close info and clear temp in lock rcd
getRecNum
curr lk rcd num = -1?
YES
NO
get rcd num from Lock Table database
updateLockRec
update lock rcd lastdate = cls time seq cls tries = 0
Update lock file
updateBackupRec
Update lock backup file
update okay?
YES
NO
B
P 13C

FIG. 44B

Cencon 2000 for Windows 95
Covert Entry (Cover) Close Code
Flowcharts
Close a Route Using a Menu

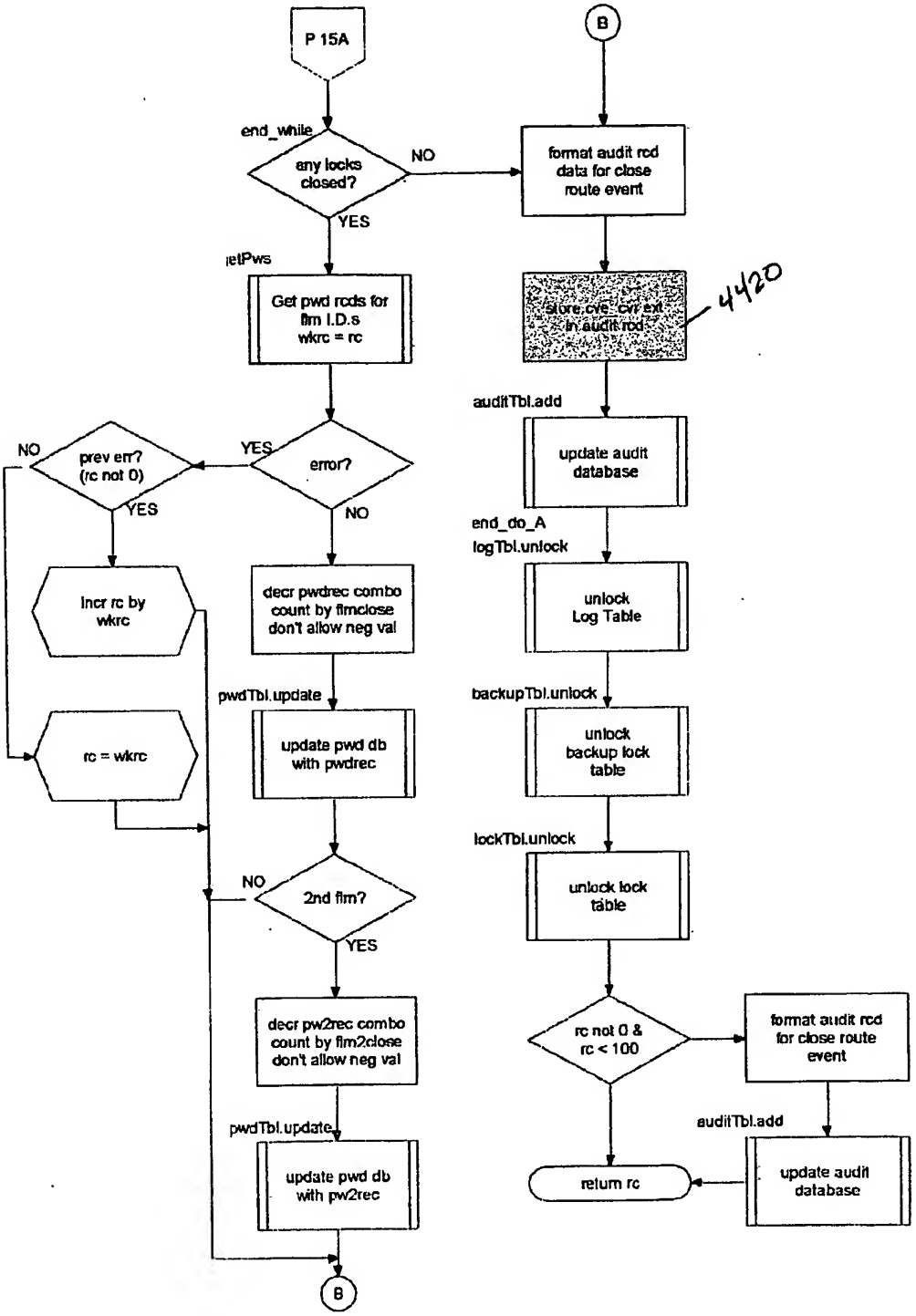


FIG. 44c

Cencon 2000 for Windows 95
Covert Entry (Cover) Close Code
Flowcharts
Close a Route Using Keys

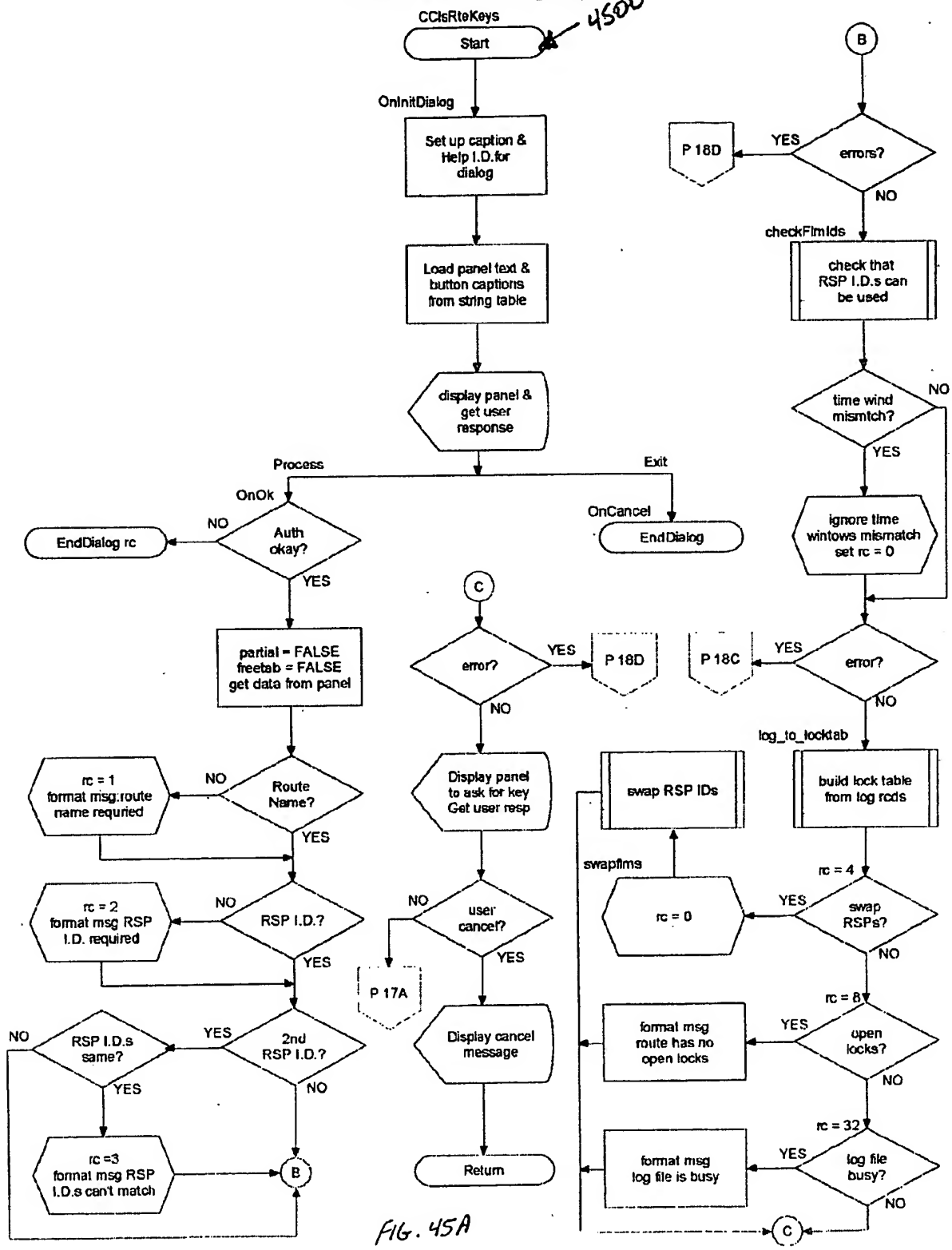


FIG. 45A

Cencon 2000 for Windows 95
Covert Entry (Cover) Close Code
Flowcharts
Close a Route Using Keys

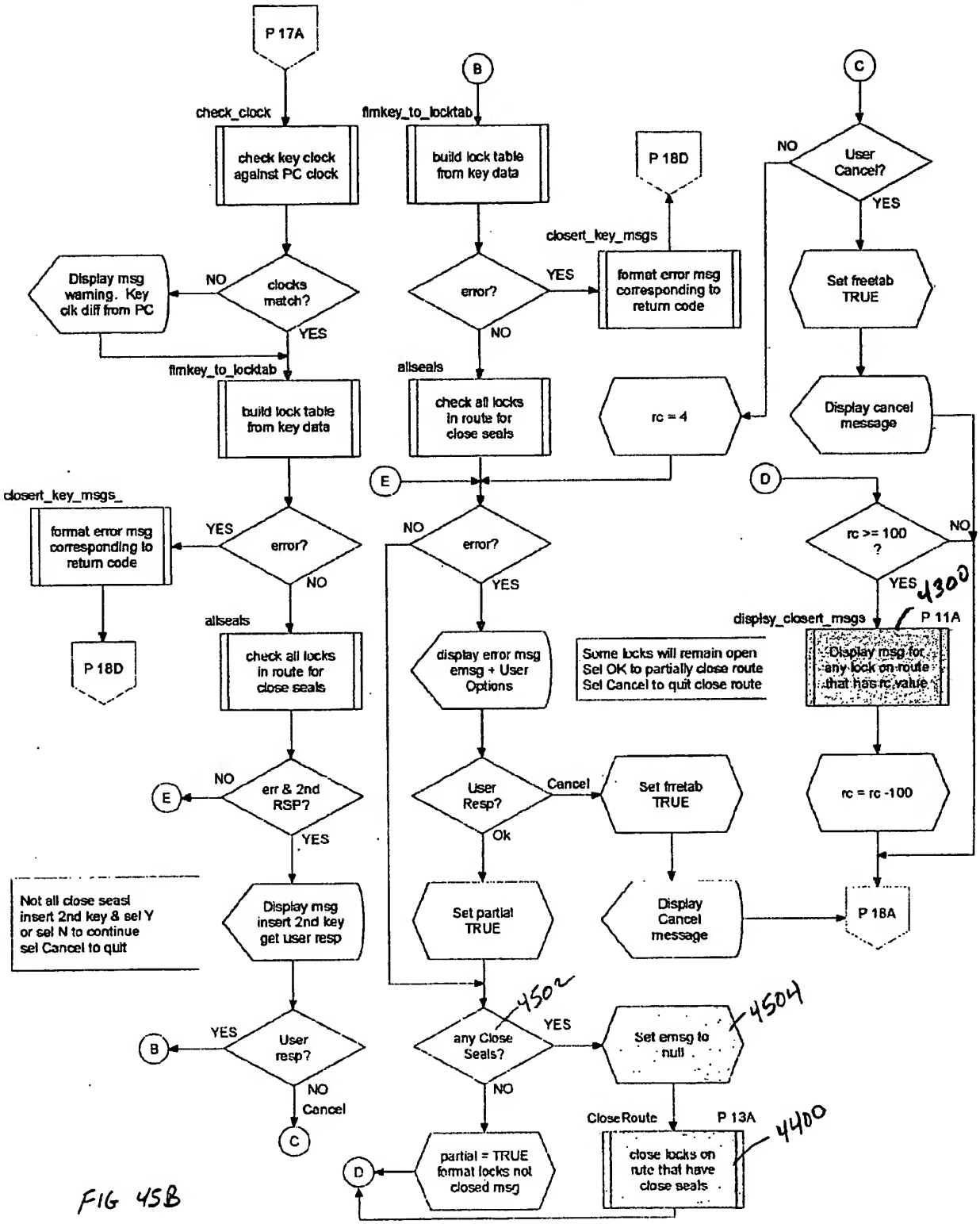


FIG 45B

Cencon 2000 for Windows 95
Covert Entry (Cover) Close Code
Flowcharts
Close a Route Using Keys

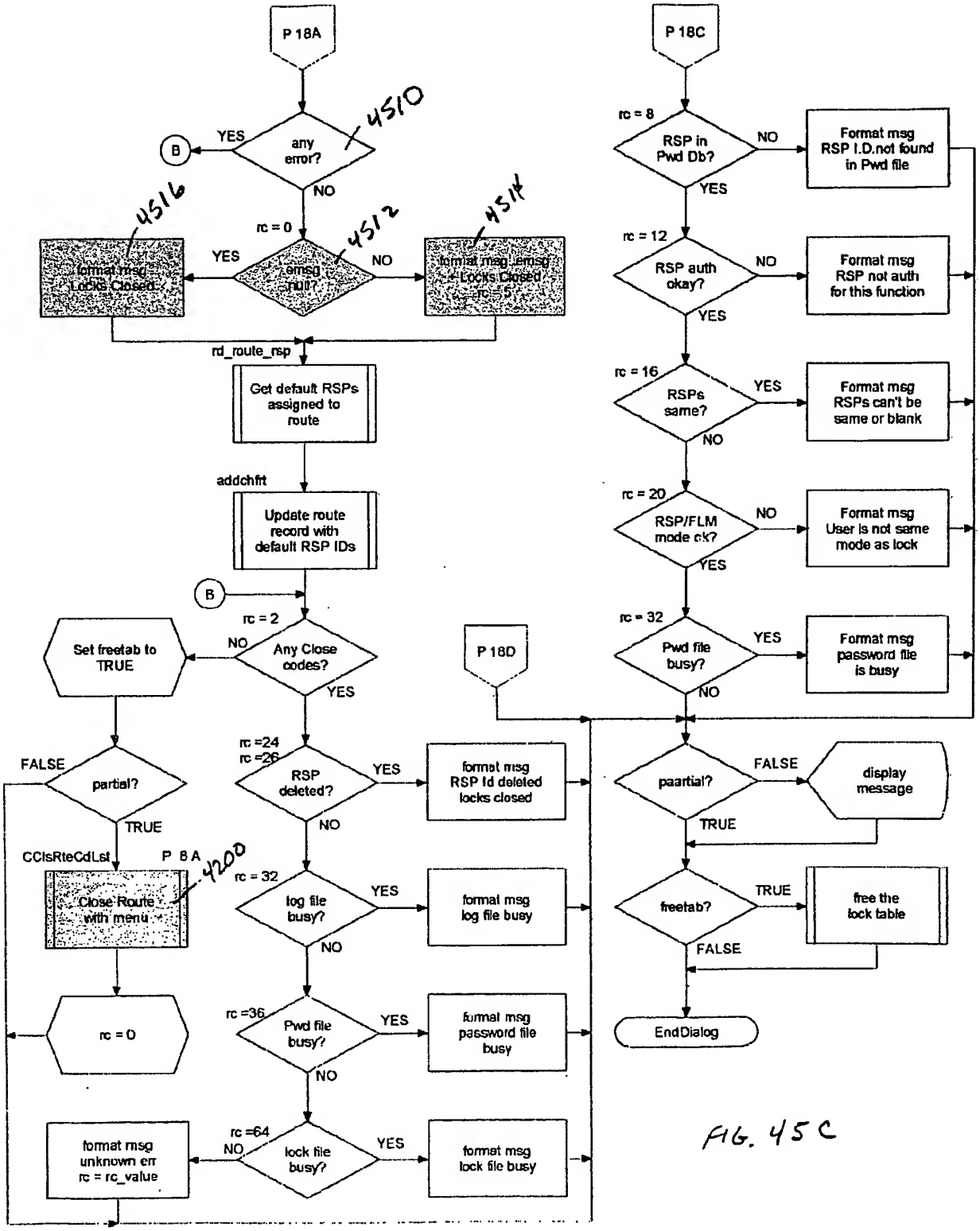


FIG. 45C

Cencon 2000 for Windows 95
Covert Entry (Cover) Close Code
Flowcharts
Check for CVE(Cover) Close Code

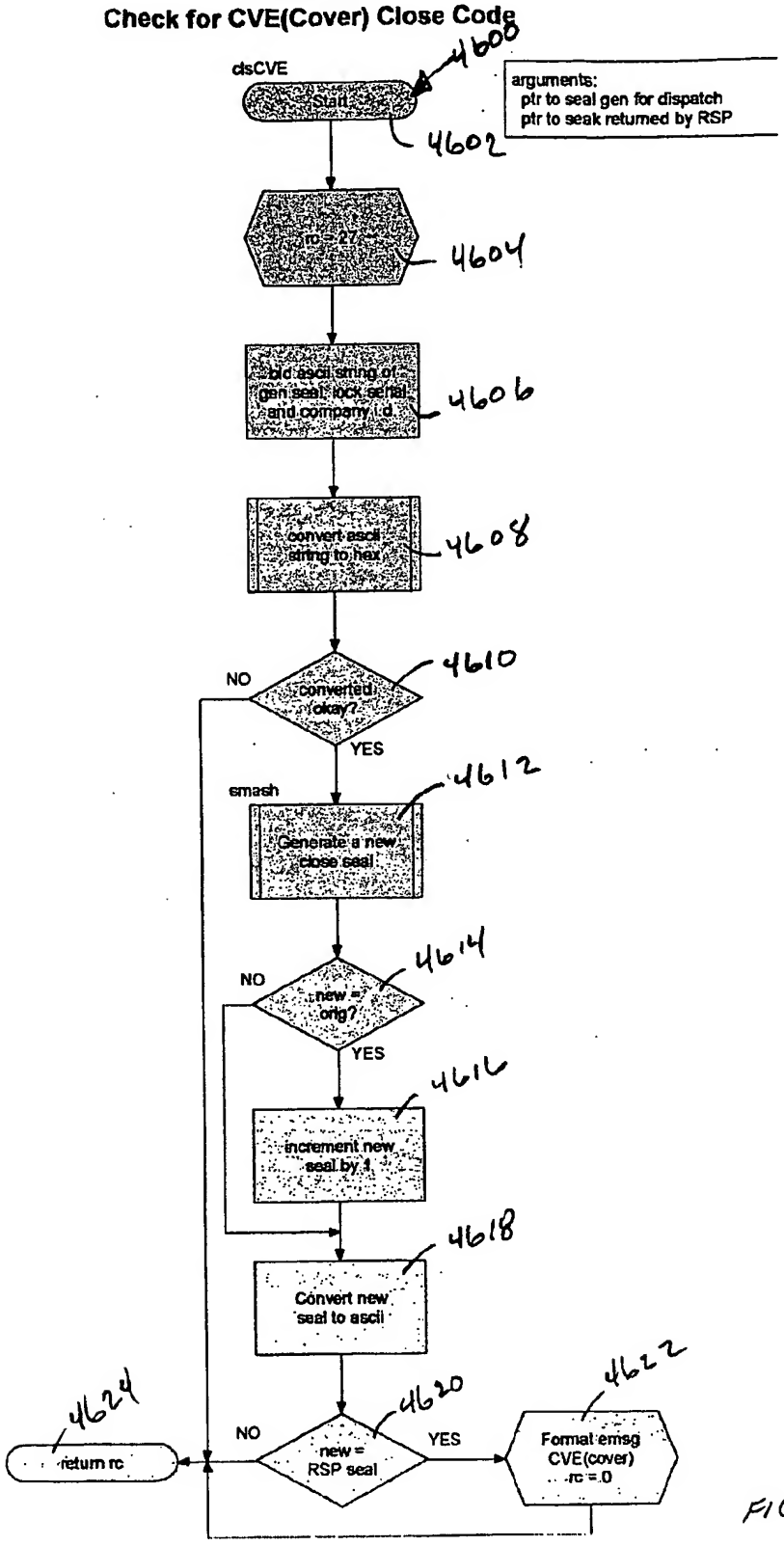


FIG 46

INSERTS INTO FILE MAINLINE.ASM

4700

```
PROC MAIN
~~~~~~~~~~~~
POR:
  Clear flag for supershelve pass 2 (SS_pass2 = 0).
  Clear housekeeping record data read flag (HK_flg = 0).
WDTO:
  {{{current 72M code unchanged to label pwr_lop:}}}
pwr_lop:
  DO
    Check for a covert bolt movement (call to chk_bolt_covert).   4800
  UNTIL (VCAP has a 5 volt charge (n5_0volt = 0))
  {{{current 72M code unchanged to label Fact_Vfy:}}}
Fact_Vfy:
  {{{current 72M code unchanged until call to EEhk_Get}}}          5000
  Read the housekeeping record data into RAM (call to EEhk_Get).
  Set the housekeeping record data read flag (HK_flg = 1).
  {{{current 72M code unchanged to label kcklop:}}}
kcklop:
  DO
    Check for a covert bolt movement (call to chk_bolt_covert).   4800
    IF (watchdog time-out occurs (Wdog_flg = 1))
      Restart lock firmare at label WDTO.
    ENDIF
  UNTIL (kicker capacitor charged (nKick_Det = 0))
  IF (bolt was retracted (bolt_opn = 1) .AND. bolt has not been closed yet
     (bolt_cls = 1))
    Output a dial left "dL" message to the LCD.
    Flag that a complete dial turn in one direction has not been done
       (CAM_OK = 1).
    Enable dial pulse interrupts.
    DO
      Wait in low power state for a processor interrupt.
      Check for a covert bolt movement (call to chk_bolt_covert).   4800
    UNTIL (a complete dial turn in one direction has been done
            (CAM_OK = 0))
    Flag that bolt has been closed (bolt_cls = 0).
    Write bolt flag to EEPROM housekeeping record.   4900
  ENDIF
wait4_cnt:
  Enable dial pulse interrupts.
pwr_lop1:
  DO
```

FIG. 47A

```
    Check for a covert bolt movement (call to chk_bolt_covert).   -4800
  UNTIL (VCAP has charged to 5 volts (n5_0volt = 0))
  Wait for a further half turn of the dial in any direction (call to Wait4_haf)).
chrg_lop:
  DO
    Wait in low power state for a processor interrupt.
    Check for a covert bolt movement (call to chk_bolt_covert).   -4800
      IF (watchdog time-out occurs (Wdog_flg = 1))
        Restart lock firmare at label WDTO.
      ENDIF
  UNTIL (enough fast enough pulses have been detected to charge the
          kicker capacitor (Chg_Ctr > 0))
  {{{current 72M code unchanged to label no_silent:}}}
no_silent:                                                    5200
  {{{current 72M code unchanged until call to Chk4_Open}}}
  IF (a covert entry was detected (CE_flg = 1))
    Display "CE" message on the lock LCD.
    Wait for a half turn in any direction (call to Wait4_haf).
    Display "ECE" prompt on the LCD for entry of the covert clearing
      combination.
no_covert:
  ELSE
    Display "EC" prompt on the LCD for entry of the normal combination.
  ENDIF
bypass:
  {{{current 72M code unchanged to label Main_lop:}}}
Main_lop:
  DO
    Wait in low power state for a processor interrupt.
    Disable interrupts.
    Check for a covert bolt movement (call to chk_bolt_covert).   -4800
    IF (a keystroke was entered (key_down = 1)
      Clear the keystroke entered flag (key_down = 0).
      Output a beep (call to Sound_beeper).
    ENDIF
    Enable interrupts.
  UNTIL (a pair of keystrokes is entered on the lock keypad (pair_is_in = 1)
                .OR. watchdog time-out occurs (Wdog_flg = 1)}
  {{{current 72M code unchanged to end of the routine}}}
These routines are called in part of the code:
Call to ChgKyIn  -  5800
Call to Combo_in  -  5100
Call to Wait4_TM  -  5700
ENDPROC
```

FIG. 47B

4800

```
PROC CHK_BOLT_COVERT
chk_bolt_covert:
  IF (Bolt power supply is ON (bolt_ps = 1))
cov_1:
    Reset watchdog timer.
    IF (Housekeeping record data not in RAM (HK_flg = 1)
      Read housekeeping record data into RAM.   — 5000
      Set housekeeping record read flag (HK_flg = 1).
    ENDIF
cov_2:
    IF (Backcover was removed (BkCvr_Sw = 0))
      Clear the user's smartkey serial number for the audit record.
      Clear the time stamp for the audit record.
      Set the covert entry flag (CE_flg = 1).
      Set the backcover removed flag (bkcv_flg = 1).
      Write a "Backcover Removed" audit record to EEPROM.   — 4900
      Write the covert flags to the EEPROM housekeeping record. — 4900
      IF (error writing data to the EEPROM)
        Output an error indication to the LCD and restart lock firmware.
      ENDIF
cov3:
      DO
        Latch the power supply.
cov_4:
      UNTIL (Bolt power supply is OFF for 25 ms (bolt_ps = 0))
    ELSE
cov_5:
      DOWHILE (Bolt power supply is ON (bolt_ps = 1)
        IF (Bolt direction is opening (bolt_ext = 0))
          Set bolt direction flag to opening (bolt_flg = 0).
          Set bolt opened flag (bolt_opn = 1).
        ELSE
cov_6:
          Set bolt direction flag to closing (bolt_flg = 1).
        ENDIF
cov_7:
        IF (Lock opened by combination entry (bolt_cls = 1))
          Save the bolt flags (bolt_flg and bolt_open) to the EEPROM  — 4900
            housekeeping record.
        ELSE
cov_8:
          Clear user's smartkey serial number for the audit record.
          Clear the time stamp for the audit record.
          Set the covert entry flag (CE_flg = 1).
```

FIG. 48A

```
        Write a covert entry audit record to the EEPROM.  - 4900
        Save the covert entry flag (CE_flg) to the EEPROM housekeeping  - 4900
          record.
        IF (Audit Record Write Error)
          Output error code and restart firmware.
        ENDIF
      ENDIF
debounce:
      Latch lock power.
      Set flags to wait for a full dial turn.
debounce1:
      DO
        IF (Full dial turn detected)
          Mark bolt as closed (bolt_cls = 0).
          Save bolt closed flag to the EEPROM housekeeping record.  - 4900
        ENDIF
cov_10 through cov_15:
      UNTIL (Bolt supply turns OFF and stays off for 25 ms (bolt_ps = 0)
            .OR. bolt was opening but is now closing for 25 ms
            (bolt_flg = 0 and bolt_ext = 1) .OR. bolt was closing but is
            now opening for 25 ms(bolt_flg = 1 and bolt_ext = 0))
    ENDDO
  ENDIF
cov_16:
  Clear processor good restart flags (Ini_flgs = 0))
  IF (VCAP voltage not still good (n3_2volt = 1))
    Unlatch power.
    DOWHILE (.TRUE.)
      Wait for VCAP power to exhaust itself.
    ENDDO
  ELSE
cov_17:
    Enable knockoff release (KO_Rel = 1).
cov_18:
    Perform knockoff operation.
    Disable knockoff release (KO_Rel = 0).
    Restart lock firmware at power on reset address.
  ENDIF
ENDIF
cov_19:
RETURN
ENDPROC
```

FIG. 48B

INSERTS INTO FILE I2C_EE65.INC

4900

```
PROC EEHK_SAV
~~~~~~~~~~~~~~~~
EEhk_Sav:
  {{{current 72M code unchanged until just prior to call to SecEE_Writ}}}
  Move covert entry flags byte to output record buffer.
  {{{current 72M code unchanged to end of routine}}}
  RETURN
ENDPROC
```

FIG. 49

5000

```
PROC EEHK_GET
~~~~~~~~~~~~~~~~
EEhk_Get:
  {{{current 72M code unchanged until just prior to end of routine}}}
  Move covert entry flags byte from input record buffer to dedicated RAM.
  RETURN
ENDPROC
```

FIG. 50

INSERTS INTO FILE COMBO_IN.INC

5100

PROC COMBO_IN

```
Combo_in:
  {{{current 72M code unchanged to label chk4_dual:}}}
This routine is called in this part of the code:
 Call to Gen_Finl — 5500
chk4_dual:
  {{{current 72M code unchanged until LJMP to INIT_MAIN}}}
cov_chk:
  IF (a detected covert entry has not been cleared (CE_flg = 1))
    Clear the covert entry flag (CE_flg = 0).
    Clear the backcover removed flag (bkcv_flg = 0).
    Write a covert entry cleared audit record to the EEPROM.
  ENDIF
no_clear:
  IF (factory mode combination entered)
    Set factory mode open flag (Fac_Opn = 1).
    Flag bolt as not closed yet (bolt_cls = 1).
    Write opening flags to the EEPROM housekeeping record. — 4900
    Set flag to enable a lock bolt open (ok_to_open = 1).
  ELSE
Un-Lock:
    Set flag to enable a lock bolt open (ok_to_open = 1).
    Save the smartkey serial number of the last user to enter a
combination.
    Save the data pertinent to the lock opening (call to Sav_Open). — 5600
  ENDIF
Alow_Lock:
  {{{current 72M code unchanged to label wait4cls:}}}
wait4cls:
  DOWHILE (CAM switch is open (nCAM_Sw = 1)
    Wait in low power state for a processor interrupt.
    Check for a covert bolt movement (call to chk_bolt_covert). — 4800
  ENDDO
wait4opn:
  DO
    Wait in low power state for a processor interrupt.
    Check for a covert bolt movement (call to chk_bolt_covert). — 4800
  UNTIL (CAM switch is opened (nCAM_Sw = 1)
  {{{current 72M code unchanged to label ko_strt:}}}
This routine is called in this part of the code:
```

Fig. 51 A

```
Call to Gen_Dly — 5400
ko_strt:
  DO
    Check for a covert bolt movement (call to chk_bolt_covert). — 4800
    Wait for 200 ms (call to DLY_200).
  UNTIL (enough emitter pulses or enough time has passed to perform the
         knockoff operation)
  {{{current 72M code unchanged to label ccw_lop:}}}
ccw_lop:
  DO
    Wait in low power state for a processor interrupt.
    Check for a covert bolt movement (call to chk_bolt_covert). — 4800
  UNTIL (8 consecutive emitter pulses in a counter-clockwise direction
         have been detected (CW_Dir = 0) .OR. Watchdog time-out occurs
         (Wdog_flg = 1))
  Clear flags indicating a good processor reset (Ini_flgs = 0).
  Restart lock firmware at the power on reset location POR.
ENDPROC
```

FIG. 51 B

INSERTS INTO FILE CHK4_XXX.INC

52

```
PROC CHK4_OPEN
~~~~~~~~~~~~~~~~~~
chk4_open:
  IF (Lock was opened in factory mode (Fac_Opn = 1))
    IF (Bolt was not retracted (bolt_opn = 0) .OR. Bolt was retracted with
        last bolt movement being a close (bolt_opn = 1 with bolt_flg = 1))
chk_1:
      Clear the factory mode open (Fac_Opn = 0).
      Clear the bolt opened flag (bolt_opn = 0).
      Save the covert flags to the EEPROM housekeeping record.
    ELSE
      Clear the user smartkey serial number for the audit record.
      Clear the time stamp for the audit record.
      Set the covert entry detected flag (CE_flg = 1).
      Clear the bolt opened flag (bolt_opn = 0).
      Write a covert entry audit record to EEPROM.
      Save the covert flags byte to the EEPROM housekeeping record.
      IF (error writing to EEPROM)
        Output error indication to the LCD and restart lock firmware.
      ENDIF
    ENDIF
    RETURN
  ELSE
    IF (Lock was opened in FLM mode (FLM__flg = 1) .OR. Lock was
        opened in route mode (Rout_flg = 1) .OR. Lock was opened in bank
        mode (Bank_flg = 1))
chk_2:
      IF (Bolt was retracted with last bolt movement being an open
          (bolt_opn = 1 with bolt_flg = 0))
chk_3:
        Clear the user smartkey serial number for the audit record.
        Clear the time stamp for the audit record.
        Set the covert entry detected flag (CE_flg = 1).
        Clear the bolt opened flag (bolt_opn = 0).
        Write a covert entry audit record to EEPROM.
        Save the covert flags byte to the EEPROM housekeeping record.
        IF (error writing to EEPROM)
          Output error indication to the LCD and restart lock firmware.
        ENDIF
chk_4:
        RETURN
      ENDIF
```

FIG. 52A

```
chk_5:
      {{{current 72M code unchanged to label go_aud:}}}
go_aud:
      Clear the bolt opened flag (bolt_opn = 0).
      IF (Backcover was removed (bkcv_flg = 1) .AND. Opener is an FLM or
         route user (Bank_typ = 0))
        Modify close seal to reflect a backcover removal covert entry (call to
           BkCvr_Seal).                                              ——— 5300
        Place modified close seal into audit record buffer.
      ENDIF
go_aud1:
      {{{current 72M code unchanged to end of the routine}}}
    ENDIF
  ENDIF
ENDPROC
```

FIG. 52B

5300

```
PROC BkCvr_Seal
~~~~~~~~~~~~~~~
BkCvr_Seal:
  Place proprietary values into data buffer.
  Smash values into a new close seal (call to smash6to3).
  IF (new close seal same as the old close seal)
    Modify new close seal to ensure that it is different from the old.
  ENDIF
  RETURN
ENDPROC
```

FIG. 53

INSERTS INTO FILE INT_HAND.INC

5400

```
PROC Gen_Dly
~~~~~~~~~~~~~~~~
Gen_Dly:
  {{{current 72M code unchanged to label gen_lop:}}}
gen_lop:
  DO
    Wait in low power state for a processor interrupt.
    Check for a covert bolt movement (call to chk_bolt_covert).   5800
  UNTIL (generator counter has been exhausted (Gen_Ctr = 0) .OR.
         Watchdog time-out occurs (Wdog_flg = 1))
  RETURN
ENDPROC
```

FIG. 54

INSERTS INTO FILE GEN_XXXX.INC

5500

```
PROC Gen_Finl
~~~~~~~~~~~~~~~
Gen_Finl:
  {{{current 72M code unchanged to $Fact_Init assembler directive}}}
  IF (covert entry detected (CE_flg = 1)
    IF (opener is a bank mode user (Bank_typ = 1) .AND. Bank mode is not
        the only mode initialized for the lock)
      Output an error indication to the LCD and restart lock firmware.
    ELSE
      IF (opener is an FLM or route user (Bank_typ = 0))
gen_1:
        Smash old combination value into a new combination (call to
          smash6to3).
      ENDIF
    ENDIF
  ENDIF
gen_2:
  {{{current 72M code unchanged to end of routine}}}
  RETURN
ENDPROC
```

Fig. 55

INSERTS INTO FILE SAV_OPEN.INC

5600

```
PROC Sav_Open
~~~~~~~~~~~~~~~
Sav_Open:
  {{{current 72M code unchanged to label nor_sav:}}}
nor_sav:
  Flag bolt as not having been closed after combination entry
     (bolt_cls = 1).
  {{{current 72M code unchanged to end of routine}}}
  RETURN
ENDPROC
```

FIG. 56

INSERTS INTO FILE TM_IOHI.INC

5700

```
PROC Wait4_TM
~~~~~~~~~~~~~~~~
Wait4_TM:
  {{{current 72M code unchanged to label wlop:}}}
wlop:
  DO
    Wait for 1 ms (call to Dly_1ms).
    Attempt a TM key reset (call to TM_Reset).
    IF (TM key not reset)
      Check for a covert bolt movement (call to chk_bolt_covert).   -4800
      IF (Watchdog time-out occurred (Wdog_flg = 1))
        Restart firmware at label WDTO.
      ENDIF
    ENDIF
  UNTIL (TM key reset detected)
  {{{current 72M code unchanged to end of routine}}}
  RETURN
ENDPROC
```

FIG. 57

INSERTS INTO FILE CHANGKEY.INC

5800

```
PROC ChgKyIn
~~~~~~~~~~~~~~
ChgKyIn:
  {{{current 72M code unchanged to label wait4_chky:}}}
wait4_chky:
  DO
    IF (Watchdog time-out occurred (Wdog_flg = 1))
      Restart firmware at label WDTO.
    ENDIF
  UNTIL (Change key is removed (nChKey_IO = 1))
  Enable lock to be opened (ok_to_open = 1).
  Set the open as a factory mode open (Fac_Opn = 1).
  Flag bolt as not having been closed (bolt_cls = 1).
  Save open flags to the EEPROM housekeeping record.
  {{{current 72M code unchanged to end of routine}}}
  RETURN
ENDPROC
```

FIG. 58

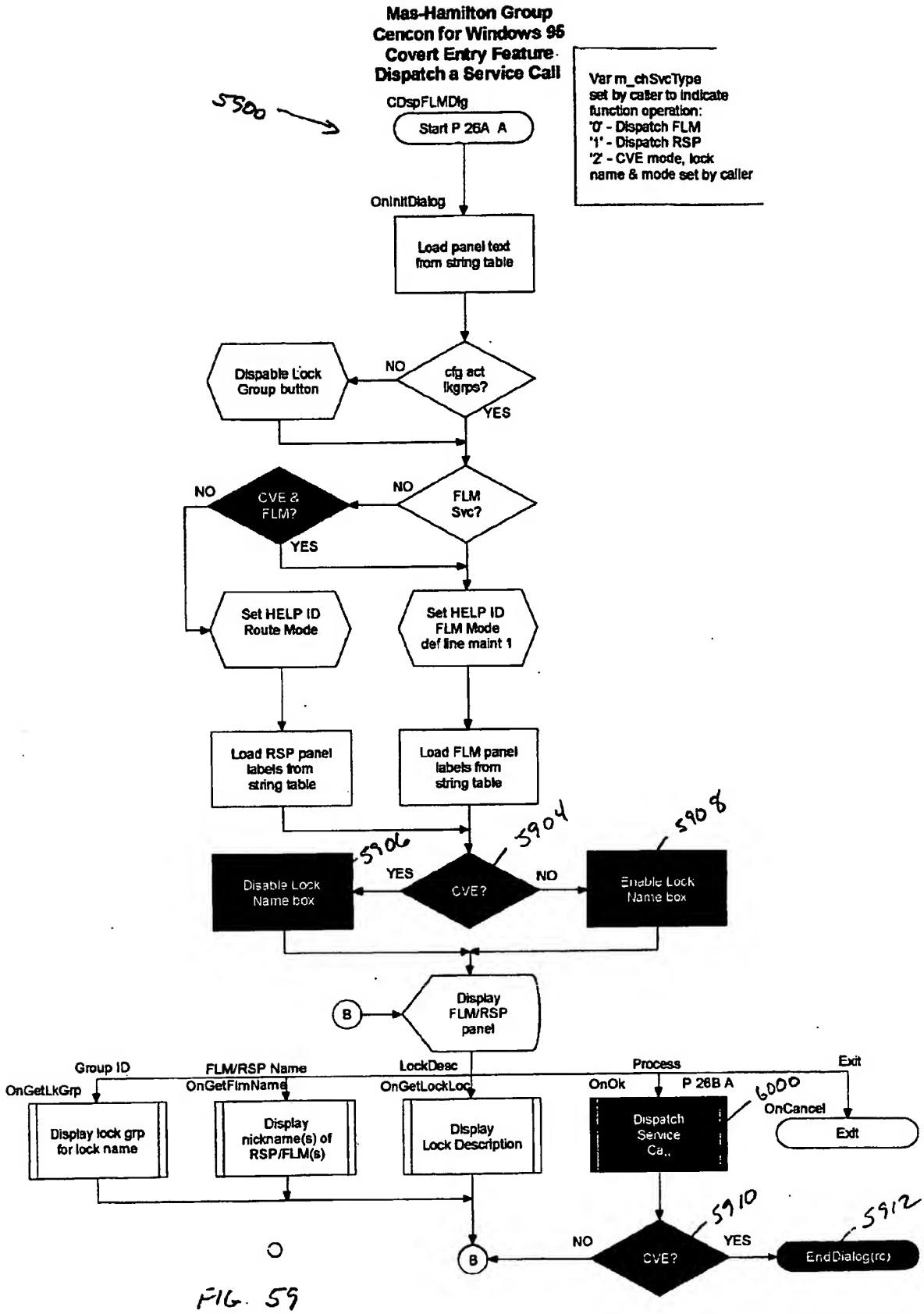

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
Dispatch a Service Call
Var m_chSvcType
set by caller to indicate
function operation:
'0' - Dispatch FLM
'1' - Dispatch RSP
'2' - CVE mode, lock
name & mode set by caller
5500
CDspFLMDlg
Start P 26A A
OnInitDialog
Load panel text
from string table
cfg act
lkgrps?
NO
YES
Dispable Lock
Group button
FLM
Svc?
NO
CVE &
FLM?
NO
YES
Set HELP ID
Route Mode
Set HELP ID
FLM Mode
def line maint 1
Load RSP panel
labels from
string table
Load FLM panel
labels from
string table
5904
5906
5908
CVE?
YES
NO
Disable Lock
Name box
Enable Lock
Name box
B
Display
FLM/RSP
panel
Group ID
OnGetLkGrp
FLM/RSP Name
OnGetFlmName
LockDesc
OnGetLockLoc
Process
OnOk
P 28B A
Exit
OnCancel
6000
Exit
Display lock grp
for lock name
Display
nickname(s) of
RSP/FLM(s)
Display
Lock Description
Dispatch
Service
Ca...
5910
5912
CVE?
NO
YES
B
EndDialog(rc)

FIG. 59

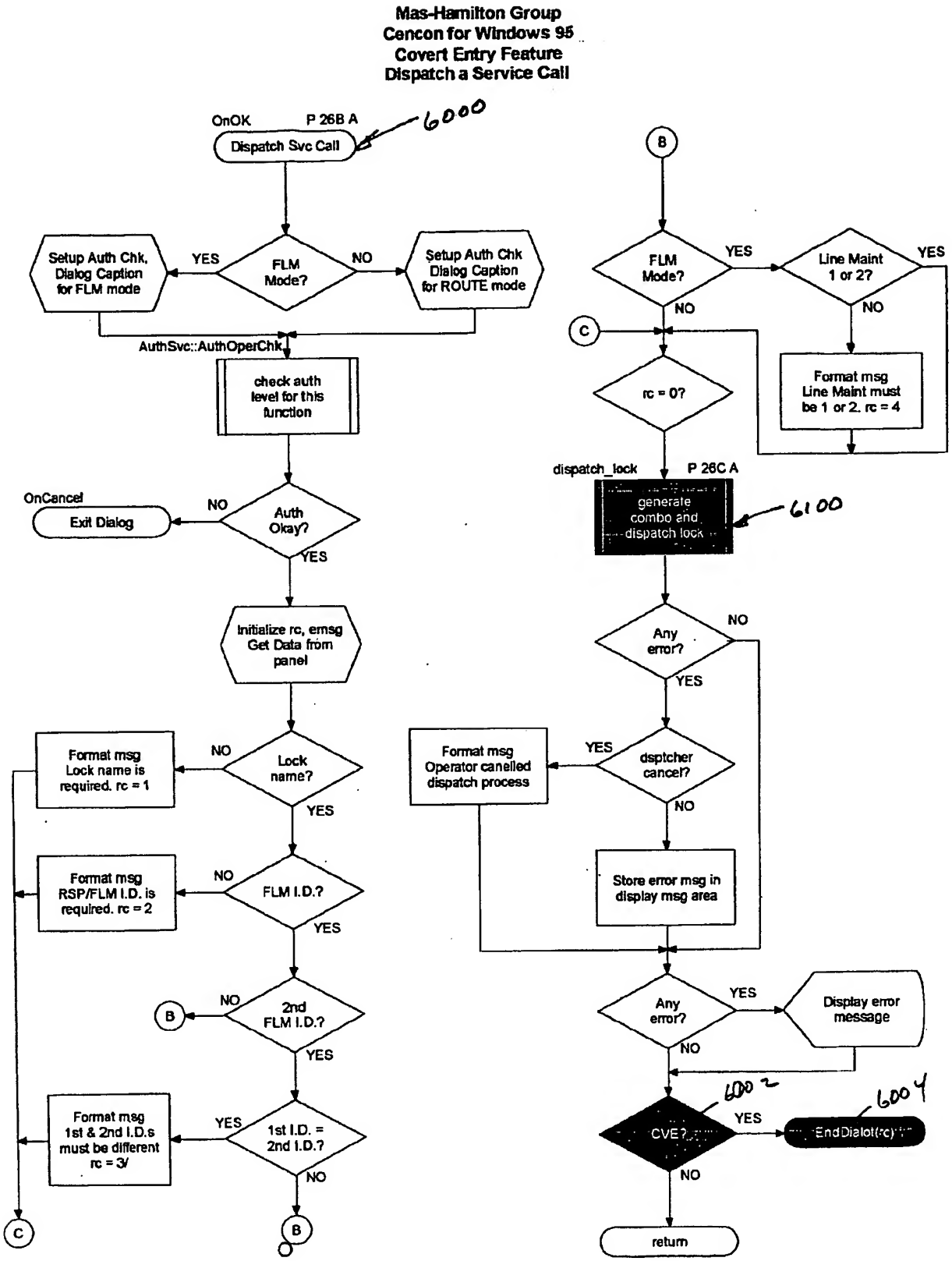

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
Dispatch a Service Call
OnOK
P 26B A
6000
Dispatch Svc Call
Setup Auth Chk, Dialog Caption for FLM mode
YES
FLM Mode?
NO
Setup Auth Chk Dialog Caption for ROUTE mode
AuthSvc::AuthOperChk
check auth level for this function
OnCancel
Exit Dialog
NO
Auth Okay?
YES
Initialize rc, emsg Get Data from panel
Format msg Lock name is required. rc = 1
NO
Lock name?
YES
Format msg RSP/FLM I.D. is required. rc = 2
NO
FLM I.D.?
YES
B
NO
2nd FLM I.D.?
YES
Format msg 1st & 2nd I.D.s must be different rc = 3/
YES
1st I.D. = 2nd I.D.?
NO
C
B
B
FLM Mode?
YES
Line Maint 1 or 2?
YES
NO
C
NO
rc = 0?
Format msg Line Maint must be 1 or 2. rc = 4
dispatch_lock
P 26C A
generate combo and dispatch lock
6100
Any error?
NO
YES
Format msg Operator cancelled dispatch process
YES
dsptcher cancel?
NO
Store error msg in display msg area
Any error?
YES
Display error message
NO
6002
6004
CVE?
YES
EndDialot(rc)
NO
return

FIG. 60

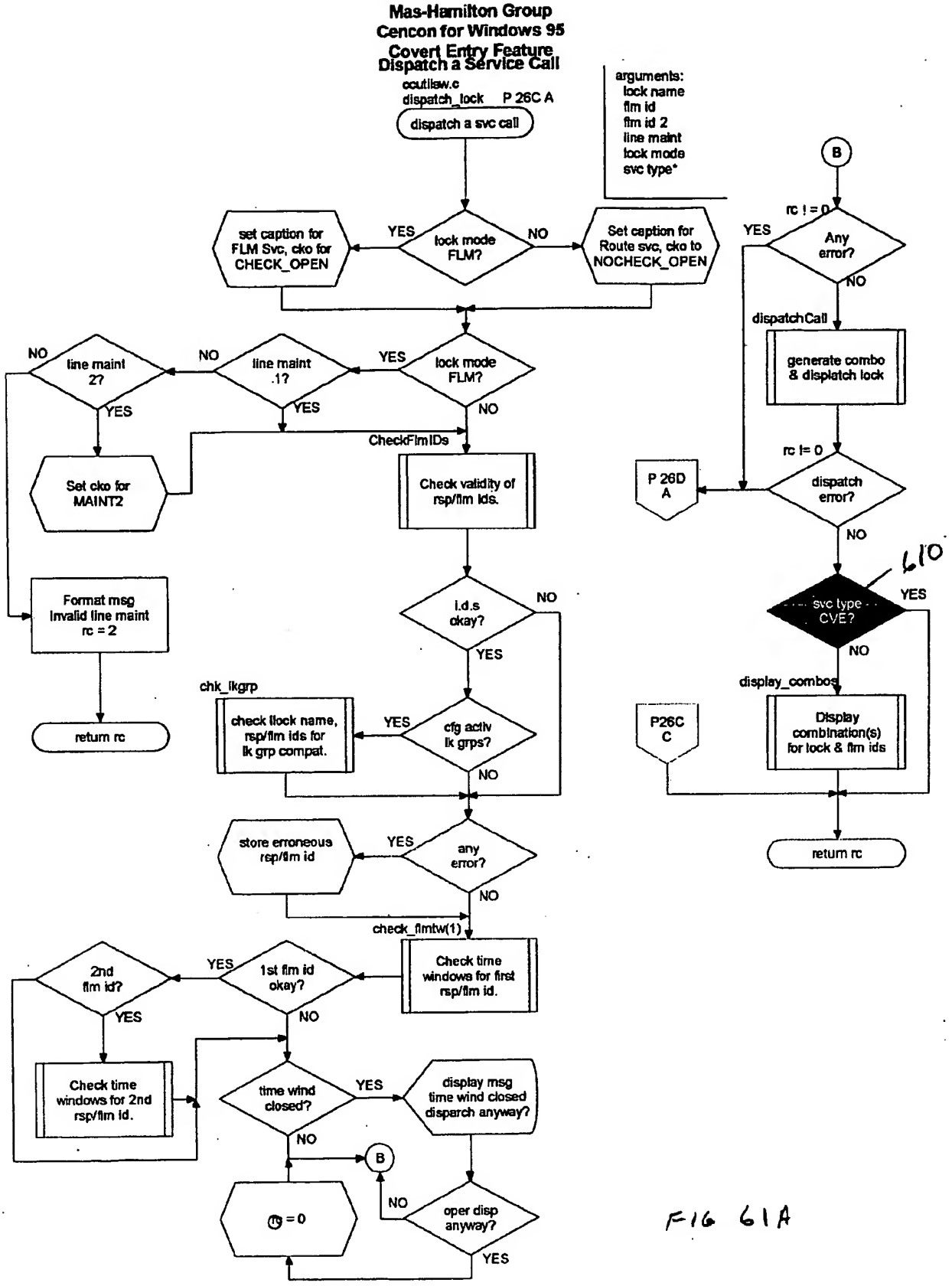

Mas-Hamilton Group
Cencon for Windows 95
Covert Entry Feature
Dispatch a Service Call
ocutilsw.c
dispatch_lock P 26C A
dispatch a svc call
arguments:
lock name
flm id
flm id 2
line maint
lock mode
svc type*
lock mode FLM?
YES
NO
set caption for FLM Svc, cko for CHECK_OPEN
Set caption for Route svc, cko to NOCHECK_OPEN
line maint 2?
line maint .1?
lock mode FLM?
Set cko for MAINT2
CheckFlmIDs
Check validity of rsp/flm ids.
Format msg Invalid line maint rc = 2
return rc
l.d.s okay?
chk_lkgrp
check lock name, rsp/flm ids for lk grp compat.
cfg activ lk grps?
store erroneous rsp/flm id
any error?
check_flmtw(1)
Check time windows for first rsp/flm id.
1st flm id okay?
2nd flm id?
Check time windows for 2nd rsp/flm id.
time wind closed?
display msg time wind closed disparch anyway?
oper disp anyway?
rc = 0
B
rc ! = 0
Any error?
dispatchCall
generate combo & displatch lock
rc != 0
dispatch error?
P 26D A
svc type CVE?
610
display_combos
Display combination(s) for lock & flm ids
P26C C
return rc

FIG 61A

INTERNATIONAL SEARCH REPORT

Int ›nal Application No
PCT/US 00/21989

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 E05B49/00

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 E05B G07C

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal, WPI Data, PAJ

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	EP 0 617 185 A (MAS HAMILTON GROUP) 28 September 1994 (1994-09-28) abstract column 5, line 5 -column 6, line 10 column 9, line 37 -column 10, line 13 column 12, line 31 -column 14, line 30 column 20, line 33 - line 47 figures 4,5 ---	1-8
Y –	US 5 488 660 A (DAWSON GERALD L ET AL) 30 January 1996 (1996-01-30) abstract column 1, line 10 - line 15 column 5, line 18 -column 6, line 62 column 13, line 21 - last line figures 1,2,7 --- -/--	1-8

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
31 January 2001	07/02/2001

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31–70) 340–2040, Tx. 31 651 epo nl, Fax: (+31–70) 340–3016	Miltgen, E

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Int onal Application No
PCT/US 00/21989

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
A	US 5 684 457 A (HARVEY MICHAEL P ET AL) 4 November 1997 (1997-11-04) abstract column 3, line 46 -column 4, line 51 figures 1,2 column 1, line 48 -column 2, line 56	1-8
A	US 5 517 184 A (MILLER J CLAYTON ET AL) 14 May 1996 (1996-05-14) column 4, line 25 - line 63	1-8

Form PCT/ISA/210 (continuation of second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Information on patent family members

Int. ...nal Application No
PCT/US 00/21989

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0617185 A	28-09-1994	US 5493279 A	20-02-1996
		CA 2116060 A	25-09-1994
		JP 7004123 A	10-01-1995
US 5488660 A	30-01-1996	CA 2133057 A	21-04-1995
		EP 0649957 A	26-04-1995
		JP 7197712 A	01-08-1995
US 5684457 A	04-11-1997	NONE	
US 5517184 A	14-05-1996	US 6052063 A	18-04-2000
		US 6034616 A	07-03-2000
		US 5847656 A	08-12-1998
		US 5777559 A	07-07-1998
		US 5973624 A	26-10-1999
		CA 2071577 A	22-12-1992
		DE 69226821 D	08-10-1998
		DE 69226821 T	21-01-1999
		EP 0519755 A	23-12-1992
		JP 2756627 B	25-05-1998
		JP 6264660 A	20-09-1994